Institute for Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3677

Recording, Compression and Representation of Dense Light Fields

Hendrik Siedelmann

Course of Study:	Informatik
Examiner:	Jun.-Prof. Dr.-Ing. Martin Fuchs
Supervisor:	Dipl.-Ing. Alexander Wender
Commenced:	2014-07-15
Completed:	2015-01-14
CR-Classification:	I.4.1, I.4.7, I.4.10, E.4

Abstract

The concept of light fields allows image based capture of scenes, providing, on a recorded dataset, many of the features available in computer graphics, like simulation of different viewpoints, or change of core camera parameters, including depth of field. Due to the increase in the recorded dimension from two for a regular image to four for a light field recording, previous works mainly concentrate on small or undersampled light field recordings.

This thesis is concerned with the recording of a dense light field dataset, including the estimation of suitable sampling parameters, as well as the implementation of the required capture, storage and processing methods. Towards this goal, the influence of an optical system on the, possibly bandunlimited, light field signal is examined, deriving the required sampling rates from the bandlimiting effects of the camera and optics. To increase storage capacity and bandwidth a very fast image compression methods is introduced, providing an order of magnitude faster compression than previous methods, reducing the I/O bottleneck for light field processing. A fiducial marker system is provided for the calibration of the recorded dataset, which provides a higher number of reference points than previous methods, improving camera pose estimation.

In conclusion this work demonstrates the feasibility of dense sampling of a large light field, and provides a dataset which may be used for evaluation or as a reference for light field processing tasks like interpolation, rendering and sampling.

Zusammenfassung

Das Konzept des Lichtfelds erlaubt eine bildbasierte Erfassung von Szenen und ermöglicht es, auf den erfassten Daten viele Effekte aus der Computergrafik zu berechnen, wie das Simulieren alternativer Kamerapositionen oder die Veränderung zentraler Parameter, wie zum Beispiel der Tiefenschärfe. Aufgrund der enorm vergrößerte Datenmenge die für eine Aufzeichnung benötigt wird, da Lichtfelder im Vergleich zu den zwei Dimensionen herkömmlicher Kameras über vier Dimensionen verfügen, haben frühere Arbeiten sich vor allem mit kleinen oder unterabgetasteten Lichtfeldaufnahmen beschäftigt.

Diese Arbeit hat das Ziel eine dichte Aufnahme eines Lichtfeldes vorzunehmen. Dies beinhaltet die Berechnung adäquater Abtastparameter, sowie die Implementierung der benötigten Aufnahme-, Verarbeitungs- und Speicherprozesse. In diesem Zusammenhang werden die bandlimitierenden Effekte des optischen Aufnahmesystems auf das möglicherweise nicht bandlimiterte Signal des Lichtfeldes untersucht und die benötigten Abtastraten davon abgeleitet. Um die Bandbreite und Kapazität des Speichersystems zu erhöhen wird ein neues, extrem schnelles Verfahren der Bildkompression eingeführt, welches um eine Größenordnung schneller operiert als bisherige Methoden. Für die Kalibrierung der Kamerapositionen des aufgenommenen Datensatzes wird ein neues System von sich selbst identifizierenden Passmarken vorgestellt, welches im Vergleich zu früheren Methoden mehr Referenzpunkte auf gleichem Raum zu Verfügung stellen kann und so die Kamerakalibrierung verbessert.

Kurz zusammengefasst demonstriert diese Arbeit die Durchführbarkeit der Aufnahme eines großen und dichten Lichtfeldes, und stellt einen entsprechenden Datensatz zu Verfügung. Der Datensatz ist geeignet als Referenz für die Untersuchung von Methoden zur Verarbeitung von Lichtfeldern, sowie für die Evaluation von Methoden zur Interpolation, zur Abtastung und zum Rendern.

Danksagung

Ich möchte mich bei Kathi bedanken, für ihr Verständnis, für ihre Rücksicht und dafür das sie da war.

Mein Dank gilt außerdem Martin Fuchs, der mir diese fantastische Möglichkeit gegeben hat, sowie Alexander Wender, der mir stets mit Rat und Tat zur Seite stand.

Auch danke ich meiner Familie, dafür dass sie mir mein Studium ermöglicht hat, und besonders Kerstin, für einen dringend benötigten Kurzurlaub und Caren fürs Korrigieren. Bedanken möchte ich mich auch bei Björn Heling, für die Korrekturen und hilfreichen Anmerkungen.

Contents

Abstract . . . 3
Zusammenfassung . . . 4
List of Figures . . . 13
List of Tables . . . 15
List of Listings . . . 15
List of Abbreviations . . . 16

1 Introduction 17
1.1 Motivation . . . 17
1.2 Objectives . . . 18
1.2.1 Primary Objective . . . 18
1.2.2 Secondary Objectives . . . 18
1.3 Contributions . . . 18
1.4 Outline . . . 19

2 Background 21
2.1 Projection and Optics . . . 21
2.1.1 Homogeneous Coordinates . . . 21
2.1.2 Perspective Projection . . . 22
2.1.3 Pose Estimation and Camera Calibration . . . 24
2.1.4 Thin Lens Model . . . 25
2.1.5 Aberrations . . . 26
2.2 Light fields . . . 28
2.2.1 Light Field Capture . . . 28
2.2.2 Spatial Multiplexing . . . 30
2.2.3 Frequency Multiplexing / Compressive Sensing . . . 30
2.3 Sampling . . . 31
2.3.1 Aliasing . . . 31
2.3.2 Sampling of Bandlimited Signals . . . 31
2.3.3 Modulation Transfer Function . . . 32
2.3.4 Resolution . . . 32
2.3.5 Bayer Pattern Sensor . . . 33
2.3.6 Light Field Sampling . . . 34
2.3.7 Spatial Bandlimit . . . 34
2.3.8 Angular Bandlimit . . . 36

2.4 Compression . . . 37
2.4.1 Entropy Coding . . . 37
2.4.2 Decorrelation . . . 39

3 State of the Art 43
3.1 Existing Light Field Datasets . . . 43
3.1.1 Sampling . . . 44
3.1.2 Conclusion . . . 45
3.2 Fast Lossless Compression . . . 45
3.2.1 Text and 1D Compression . . . 45
3.2.2 Image Compression . . . 47
3.2.3 Video Compression . . . 47
3.2.4 Lossless Light Field Compression . . . 48
3.2.5 Conclusion . . . 48
3.3 Fiducial Markers . . . 48
3.3.1 Basic Detection . . . 49
3.3.2 Error Correction . . . 49
3.3.3 Accuracy . . . 50
3.3.4 Reference Point Density . . . 50
3.3.5 Conclusion . . . 52

4 System Design 53
4.1 Overview . . . 54
4.2 Dimensions . . . 55
4.3 Mechanical Setup . . . 56
4.4 Optical Setup . . . 56
4.4.1 Camera . . . 57
4.4.2 Parameters . . . 57
4.4.3 Evaluation . . . 60
4.5 Processing and Storage . . . 61
4.6 Compression . . . 63
4.7 Calibration . . . 63

5 Implementation 65
5.1 Optics and Sampling . . . 65
5.1.1 Parameters . . . 66
5.1.2 Resolution Simulation and Measurements . . . 66
5.1.3 Spatial Sampling . . . 67
5.1.4 Angular Sampling . . . 68
5.1.5 Bayer Pattern Sampling . . . 68
5.2 Mechanical Setup . . . 69
5.2.1 Power and Control . . . 70

5.2.2 Arm . . . 70
5.2.3 Turntable . . . 71
5.3 Scene . . . 72
5.3.1 Illumination . . . 73
5.3.2 Contents . . . 73
5.4 Capture and Processing Hardware . . . 73
5.4.1 Camera . . . 74
5.4.2 Transmission and Storage . . . 75
5.5 Compression . . . 76
5.5.1 Overview . . . 76
5.5.2 1D Predictor . . . 78
5.5.3 Modulo Delta Coding . . . 79
5.5.4 Frequency Substitution . . . 79
5.5.5 Significant Bit Count . . . 80
5.5.6 Block Wise Interleaved Bitpacking . . . 80
5.5.7 Recursive Application . . . 81
5.5.8 Inter Coding . . . 82
5.6 High Density Fiducial Markers for Metric Calibration . . . 83
5.6.1 Marker Structure . . . 83
5.6.2 Detection and Identification . . . 88
5.7 Software . . . 94
5.7.1 File Format . . . 94
5.7.2 Recording . . . 95
5.7.3 Calibration . . . 95
5.7.4 Rendering . . . 95
5.7.5 Motor Control . . . 98
5.7.6 Auxiliary Tools . . . 98

6 Results and Limitations 101
6.1 Compression . . . 101
6.1.1 Evaluation Method . . . 101
6.1.2 Compression Performance . . . 102
6.1.3 Archival . . . 103
6.2 Calibration . . . 105
6.2.1 Marker Detection . . . 105
6.2.2 Camera Calibration . . . 106
6.2.3 Calibration Bias . . . 107
6.3 Light Field Dataset . . . 109
6.3.1 Synthetic Aperture Rendering . . . 109
6.3.2 Viewpoint Sampling . . . 111
6.3.3 Enhanced Resolution Rendering . . . 112
6.3.4 Bayer Pattern Sampling . . . 113

6.3.5 Signal Quality 114

7 Discussion and Future Work 115
7.1 Summary 115
7.2 Perspectives on Light Field Sampling 116
7.3 Applications to Dense Sampling 116
7.4 On Resolution and Sampling 117
7.5 Fast Lossless Image Compression 118
7.6 High Density Fiducial Markers 119

8 Conclusion 121

A Appendix 123
A.1 Documentation of Tools 123
A.1.1 lfbrowser 123
A.1.2 lfextract 123
A.1.3 lfrecord 124
A.1.4 lfavg 124
A.1.5 lfinfo 125
A.2 MTF Plots 126
A.3 Contents of the Recorded Light Field 128

Index 131

Bibliography 133

List of Figures

2.1 perspective projection . . . 23
2.2 thin lens model . . . 24
2.3 focus blur in the thin lens model . . . 25
2.4 two plane light field parametrisation . . . 29
2.5 sphere angle light field parametrisation . . . 29
2.6 example Bayer pattern . . . 33
2.7 dimensions of a light field . . . 34
2.8 bandlimiting effects in the thin lens model . . . 35
2.9 MTF plot of angular light field components . . . 36
2.10 Bayer pattern packing . . . 40

3.1 plot of bandwidth versus compression ration for lossless compression methods . 46
3.2 `aruco` marker detection depending on marker size . . . 51

4.1 hardware setup . . . 53
4.2 cross section of the optical setup . . . 58
4.3 depth of field and scene diameter against focal length . . . 60
4.4 optimizing RAID bandwidth . . . 62

5.1 MTF simulation and measurements . . . 67
5.2 components of the mechanical setup . . . 70
5.3 components of the turntable . . . 71
5.4 scene overview . . . 72
5.5 inverse camera response . . . 74
5.6 flow chart of the compression procedure . . . 77
5.7 prediction and bitpacking . . . 78
5.8 triangle function used for frequency substitution . . . 80
5.9 interleaved bitpacking . . . 81
5.10 components of a single marker . . . 84
5.11 multi marker layout . . . 85
5.12 paging of marker grids . . . 85
5.13 irregularization of marker and page id . . . 87
5.14 alternative marker sizes . . . 87
5.15 flow chart of the marker detection process . . . 88

5.16 checkerboard corner score 89
5.17 corner score parameters 90
5.18 corner template 91
5.19 marker template 92
5.20 focus blur in synthetic aperture rendering 96

6.1 plot of compression ratio versus group of picture size 102
6.2 plot of bandwidth versus compression ratio 103
6.3 conversion time versus file size 104
6.4 marker detection example 105
6.5 visualization of combined mechanical and calibration precision 106
6.6 reprojection error visualized over the calibrated image 108
6.7 synthetic aperture rendering 110
6.8 large aperture capture 111
6.9 enhanced resolution rendering 112
6.10 aliasing in the red and blue channel 113
6.11 demonstration of the dynamic range 114

A.1 mtf plot of a simulated lens at $f/16$ 126
A.2 mtf plot of the used lens at $f/16$ 127
A.3 scene contents - large compartment 128
A.4 scene content - small compartments 129

List of Tables

4.1 properties of camera and sensor 57
4.2 scene parameters 61

5.1 characteristic of the used lens 65

6.1 parameters of the recorded light field 109

List of Listings

5.1 marker coding 86
5.2 page coding 86

List of Abbreviations

ANS	Asymmetric numeral system
BBP	block wise bitpacking
C/P	cycles per pixel
CoC	circle of confusion
DOF	depth of field
FOV	field of view
GOP	group of picture
HDR	high dynamic range
Lp/mm	line pairs per millimeter
Lp/mm	line pairs per millimeter
MTF	modulation transfer function
PnP	perspective-n-point
PSF	point spread function
RGB	red green blue
RMS	root mean square
SIMD	single instruction multiple data

1 Introduction

The theory of light fields, based on the plenoptic function, which describes the radiance in a space as a function of position and direction [Ger36, AB91], gives, in its 4D form, rise to image based rendering [LH96, GGSC96]. Image based rendering allows the simulation of many effects otherwise only available trough specialized optics, like large apertures or focal plane tilt [LH96, SYGM03, VGT+05], with few restrictions on the lens parameters. Other applications without equivalent in real optics include the change of viewpoint, or vision trough partial obstacles [LCV+04, WJV+05]. All this can be achieved without the construction of a geometric model and without costly simulation of light propagation, as is necessary in computer graphics. A captured light field contains all optical effects of a scene in a recorded form, allowing for simple rendering of complex optical effects using only lookup and averaging of recorded samples. Additional applications of light fields include geometric scene reconstruction or real 3D displays, which providing more than only stereo view, by adapting to the position of the viewer in space. However, compared to traditional 2D image capture, light field capture increases the number of dimensions necessary for capture from 2 to 4, with the according increase in the amount of sheer data that has to be captured and processed. For example, one way to capture 4D light fields is to take images with a regular 2D camera from a 2D array of viewpoint positions, multiplying the amount of data that is captured for rendered output images of similar resolution.

1.1 Motivation

Due to the large space and processing requirements of light fields, previous works concentrated on low resolution and/or undersampled light fields to keep the amount of data low, see Section 3.1. However, this limits the envelope of image based rendering, as either viewpoints are concentrated within a small space, reducing the parameters of image based rendering, like aperture and viewpoint change, or otherwise large gaps have to be left between viewpoints, which can lead to double images on rendering. While viewpoint interpolation allows the derivation of additional viewpoints [GZC+06], this comes with its own problems as it requires a solution to the correspondence problem [SHK+14], by matching corresponding points in two or more images, which may not be unambiguous for scenes which include reflections or complex occlusion.

A dense light field dataset could foster research in the area of light field sampling, and may be used to improve the more practical undersampling approaches to light field capture, by providing a dataset which provides ground truth for advanced viewpoint interpolation and lexica for compressive sensing, see Section 2.2.3.

1.2 Objectives

This work is concerned with the capture of a dense light field, which, due to the large amount of data, requires on the fly compression for efficient capture and processing, as well as special consideration with regards to storage and representation, to achieve fast rendering.

1.2.1 Primary Objective

The primary objective of this work is the capture of a dense light field, which includes the examination of the definition of dense sampling. While it is clear that more samples result in a more dense dataset, it seems that literature gives no reference as to what represents sufficiently dense sampling for practical light field capture. Therefore the sampling characteristics of the optical setup have to be examined, designing the capture setup accordingly.

1.2.2 Secondary Objectives

Secondary objectives, and paving the way for dense light field capture, are the evaluation of compression methods, to decrease the amount of raw data that needs to be stored and handled. Additionally the storage and representation of the light field dataset has to be handled in a way which allows efficient retrieval for rendering and other processing tasks.

1.3 Contributions

Three distinct contributions are presented in this work:

For sampling of bandunlimited light fields, the bandlimiting effects of the optical system are examined in Section 2.3.6, showing how appropriate sampling rates for alias free sampling can be obtained. A large light field dataset is recorded from these calculations, demonstrating the feasibility of dense sampling, and the possibility for high quality synthetic aperture rendering, without geometric reconstruction or viewpoint interpolation, see Section 6.3.

For fast lossless image compression a new method is shown, which provides more than an order of magnitude faster compression than the fastest dedicated image or video compression method, while obtaining a higher compression ratio than fast generic compression methods, see Section 5.5 and Section 6.1.

Regarding fiducial marker detection, a system is designed and implemented, which provides a higher density of reference points than previous approaches, while providing a low overhead error detection through a well defined layout, see Section 5.6. At the same time high accuracy refinement for the reference points is incorporated, without decreasing the density of packing. The system also allows a much higher number of markers to be addressed and identified, increasing the number of usable reference points.

1.4 Outline

This work is structured as follows: In Chapter 2 the fundamental principles of optics and cameras are introduced, as well as the foundations of sampling, leading to an estimate of the required samples rates for dense light field sampling. Also a brief introduction to data compression is provided. Chapter 3 gives an overview and evaluation of the current state of the art. In Chapter 4 the design for the dense light field capture system is introduced, including the reasoning for the respective design decisions and the parameters of such a system. Finally Chapter 5 details the implementation of the design and Chapter 6 shows the results achieved with the implemented system and highlights limitations. In Chapter 7 the results are discussed and possible directions for future work are revealed, while Chapter 8 contains the conclusion.

2 Background

This chapter gives a detailed overview over the fields and techniques relevant to this work. The chapter is structured by starting with the image formation process in Section 2.1 including models for optics and the calibration thereof. Section 2.2 gives an introduction to light fields, their parametrisation and capture. In Section 2.3 the problem of discrete sampling is introduced, highlighting the limitations and providing an application to light fields, using the optical models from Section 2.1 to describe the signal captured by a light field capture device. Finally, section 2.4 gives an introduction into compression, introducing common concepts and established methods.

2.1 Projection and Optics

The image formation in computer vision is normally modeled using ray optics. The following sections will give a brief overview over the basic concepts and common limitations.

2.1.1 Homogeneous Coordinates

Homogeneous coordinates [Sze10, Chapter 2] simplify work with perspective projections, as the projective transform becomes a simple matrix multiplication. In homogeneous coordinates a n-dimensional point is expressed using $n+1$ coordinates, e.g. for a 2D point $\tilde{p} = [\tilde{x}, \tilde{y}, \tilde{w}]$. Vectors that differ only by scale are considered equivalent. Conversion to and from inhomogeneous/Cartesian coordinates is possible by division with the last element, see Eq. (2.1):

$$\tilde{p} = [\tilde{x}, \tilde{y}, \tilde{w}] = \tilde{w}[x, y, 1] = \tilde{w}\bar{p} \tag{2.1}$$

The bar denotes the augmented vector, for use as homogeneous vector, e.g. $\bar{p} = (x, y, 1)$ is the augmented 2D point $p = (x, y)$. Throughout this work homogeneous vectors will be denoted using square brackets and homogeneous coordinates using a tilde. Points with $\tilde{w} = 0$ have no equivalent in non-homogeneous coordinates and are called points at infinity.

2.1.2 Perspective Projection

The perspective projection in computer graphics describes the projection of a 3D point in camera coordinates $p_c = (x, y, z)$, onto a 2D image plane by division with the z coordinate, resulting in the point $\tilde{p}_i$ in image coordinates. The origin of the camera coordinate system is called nodal point or camera center, the x and y axis are parallel to the image plane and orthogonal to each other, while the z axis is orthogonal to the first two, for more details see [Sze10, Chapter 2]. This basically simulates a pinhole camera, by tracing object rays trough the nodal point onto an image plane, where the nodal point represents the pinhole, see Fig. 2.1. Using homogeneous coordinates this projection can be expressed as a matrix multiplication, using the calibration matrix i, see Eq. (2.2):

$$\begin{aligned} \tilde{p}_i &= I \cdot p \\ \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{pmatrix} &= \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \end{aligned} \tag{2.2}$$

From $\tilde{p}_i$ the inhomogeneous image coordinates can be retrieved by division with $\tilde{w}$, see Eq. (2.1). The value of f in I determines the projection scale, and is the distance between image plane and nodal point, compare Fig. 2.1. Note that conversion of $\tilde{p}_i$ from homogeneous to non-homogeneous coordinates looses the depth information, as conventional cameras only record 2D images. The formulations above assume the same units for camera and image coordinates and identical origin. Normally the image coordinates are expressed in pixels and with the origin at the upper left corner. This can be accommodated by introducing two more parameters c_x and c_y which move the origin of the image. Scaling c_x, c_y and f can be used to achieve the desired unit conversion, see Eq. (2.3):

$$\tilde{p}_i = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} p \tag{2.3}$$

If the world coordinate system is not identical to the camera coordinate system, then a point p_w in world coordinates has to be transformed first, using the camera translation matrix T and the rotation matrix R according to Eq. (2.4), conventions follow [Sze10] and `OpenCV` [CAP+12]:

$$\tilde{p}_i = I[R|T]\bar{p}_w = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{2.4}$$

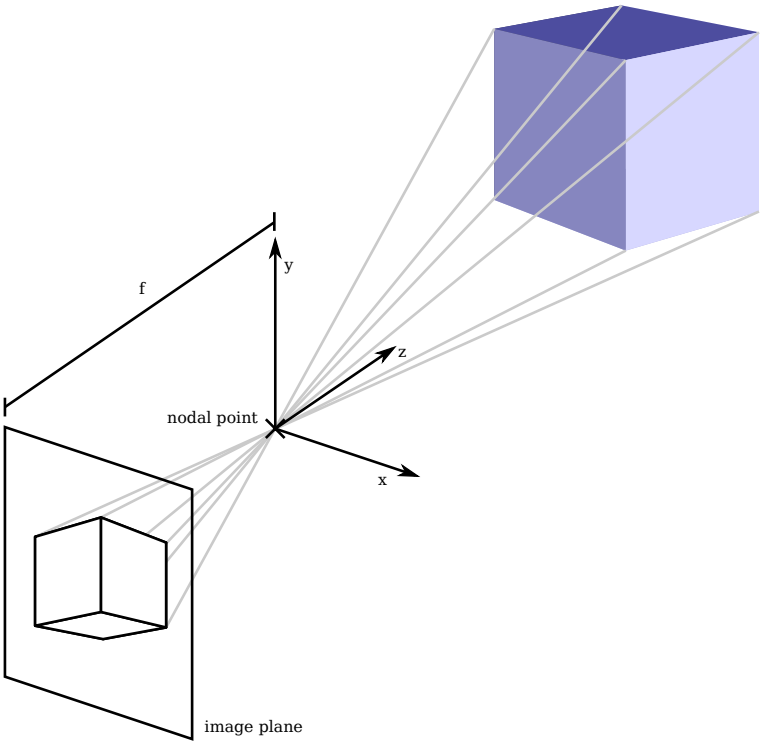


Figure 2.1: Schematic visualization of a perspective projection: A pinhole camera. The image is projected by intersecting lines trough the nodal point, or pinhole, onto the image plane. Focal length and depth translate into the distance from the image plane center while x and y position govern the direction from the center.

The matrix $M = I[R|T]$ is called camera matrix and consists of the calibration matrix I which gives the intrinsic parameters, and of the extrinsic parameters $[R|T]$. This formulation shows how intrinsic parameters are fixed for a camera, independent of orientation, while extrinsic parameters relate the camera to the world coordinates and give the camera orientation in space.

Note that R is a rotation matrix with three degrees of freedom, while translation adds another three degrees to the extrinsic parameters, for a total of six. The intrinsic calibration has, in this form, only three degrees of freedom, but intrinsic calibration normally includes additional parameters to compensate for the lens distortions, present in many real lenses, see Section 2.1.5.1.

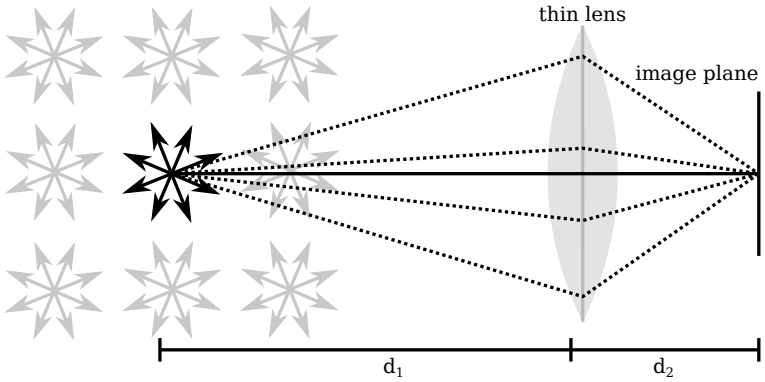


Figure 2.2: Visualization of the thin lens models. When rays from a point light source at distance d_1 intersect the thin lens, they are refracted onto the image plane at d_2.

2.1.3 Pose Estimation and Camera Calibration

In computer vision it is often mandatory to know, additional to the intrinsic parameters, the camera position and orientation for an image. If the world and image coordinates of features in the scene are known, it is possible to calculate these extrinsic parameters.

The estimation of camera orientation, given mappings from image points to world points, is known as pose estimation, extrinsic calibration or perspective-n-point problem, short PnP. The minimal case requires three points, known as perspective-3-point problem, but for increased accuracy and robustness, it is often preferred to use a higher number of points and an iterative method, minimizing the squared reprojection error, see [Sze10, Chapter 6.2]. The reprojection error e_{rms} is here defined as the distance between the observed image points, and the respective image points projected with the calibrated camera model according to Eq. (2.4). Eq. (2.5) gives the RMS, the root mean square of the reprojection error, for the corresponding image/world point pairs (x, w), used for calibration:

$$e_{rms} = \sqrt{\sum_i d(x_i, P(w_i))^2} \tag{2.5}$$

Here $d(x)$ is the distance metric, normally euclidean distance, and $P(x)$ is the full projection from Eq. (2.4), including final conversion to inhomogeneous coordinates.

Given a number of images with image to world mappings as above, it is also possible to solve for intrinsic and extrinsic parameters at the same time, for example with the method of Zhang [Zha00] which also uses an iterative method and minimizes the reprojection error.

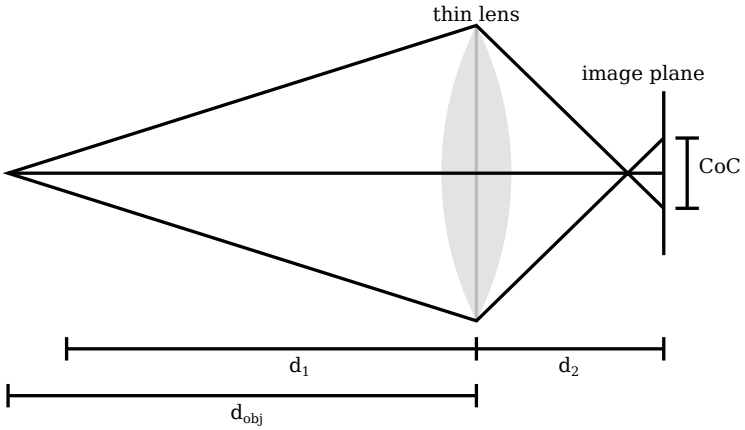


Figure 2.3: If the image plane is not at d_2 than rays originating from a point at d_1 intersect not at a single point, resulting in the point being spread onto a circle.

2.1.4 Thin Lens Model

A perspective projection describes a perfect pinhole camera, but real pinhole cameras achieve low resolution due to diffraction, see Section 2.1.5.3, and low efficiency, due to the small amount of light reaching the sensor. Most cameras are based on an optical system with one or more lenses, to achieve a projection close to that of the optimal theoretical pinhole camera. Following [Sze10], a thin lens is characterized by its focal length and the aperture of the lens. A converging lens focuses parallel rays of light onto a focal point. The distance between lens and focal point is then equivalent to the focal length. The plane trough all focal points of a lens is the focal plane. Rays that are not parallel but originating from a single point are focused according to the thin lens formula:

$$\frac{1}{f} = \frac{1}{d_1} + \frac{1}{d_2} \tag{2.6}$$

Here f denotes the focal length, and d_1 is the distance of the ray origin from the lens, also called front focal length. The back focal length d_2 is the distance from the lens to the respective focal point on the image plane. A ray which intersects the lens at the optical axis, called the chief ray, passes trough without being refracted, all other rays get refracted, fulfilling Eq. (2.6), see Fig. 2.2.

The magnification of the lens describes the reproduction scale at which objects are projected onto the image plane, and is given by the ratio of the object distance d_1, and image plane distance d_2 in Eq. (2.7), compare Fig. 2.2:

$$m = \frac{d_2}{d_1} \tag{2.7}$$

If an object is not at the focus distance given by Eq. (2.6), then the circle of confusion, short CoC, is the circle within which the rays originating from the point source intersect with the image plane. For the thin lens model the CoC can be derived geometrically using the theorem of intersecting lines, see Fig. 2.3. The full formula for the circle of confusion on the sensor, incorporating the lens magnification is given in Eq. (2.8), where a is the diameter of the aperture and d_{obj} is distance from the object to the lens:

$$c = a\frac{|d_{obj} - d_1|f}{d_{obj} \cdot (d_1 - f)} \tag{2.8}$$

For a given camera configuration and an acceptable circle of confusion, the depth of field, short DOF, is the distance between minimal and maximal d_{obj} where the CoC size falls within the acceptable limit.

2.1.5 Aberrations

The term “thin” in thin lens hints at the presumed simplification of assuming an infinitely thin lens. Real lenses are not thin and in fact normally consist of multiple lenses combined into an complex optical system, trying to approximate the behavior of a thin lens within the respective parameters and physical limitations. The divergence from the assumed model introduces a range of aberrations, shortly introduced in the following sections, and include geometric distortion, chromatic aberration and diffraction blur.

2.1.5.1 Geometric Distortion

The most prominent aberrations are geometric distortions caused by a deviation from the perfect rectilinear projection. A common cause are stops in the system, the position of which determine the type and magnitude of the distortion, as a stop located before or after the lens causes the chief ray to not pass trough the lens at the center, but at a distance, which causes the chief ray to be refracted depending on the distance from the optical axis [JW76, Wal]. This type of distortion is radially symmetric but can be relatively complex, depending on the number and position of lens elements and stops. Radial lens distortion is normally corrected using a low order polynomial [Sze10, Chapter 2.1], and can be calibrated as part of the intrinsic camera parameters, resulting in additional parameters for the intrinsic calibration. Note that distortion may also depend on the depth, although this is commonly not incorporated [AGS11].

2.1.5.2 Chromatic Aberration

In the materials available for optics today, the refractive index, which determines the refraction of light at the material boundary, is not fixed, but depends on the wavelength. The result is that a single lens has different characteristics for different colors of light. A lens with multiple

elements can partly correct for the difference in refraction, using multiple elements with differing refractive indices. Non adequate correction results in chromatic aberrations. For example if the geometric distortion depends on the wavelength, then the same point is projected onto the image plane as a curve depending on the wavelength. This lateral chromatic aberration can usually be corrected either optically or in software, using different distortion parameters for the different color channels. On the other hand longitudinal chromatic aberration, which is the inability to focus different wavelengths on the same image plane, can be fought by stopping down the aperture, and increasing depth of field until the subject is within the depth of field at all wavelengths [Sze10, Chapter 2.2].

2.1.5.3 Diffraction

The above models work with a ray model which is not physically accurate. The wave like properties of light start to show at small apertures in the form of diffraction, which limits resolution. For a point light source, the effect of diffraction projects a so called Airy pattern onto the image sensor. The size of the Airy pattern, as the radius r to the first zero of the pattern, is given in Eq. (2.9), depending on the wavelength λ, aperture size a, and focal length f. Refer to [PW13, Chapter 11.2] for more details:

$$r = \frac{1.22\lambda f}{a} \tag{2.9}$$

2.2 Light fields

The camera models established in the previous sections, describe how light is projected onto an image plane from a three dimensional scene. But can we describe the whole scene without explicitly modeling geometry, surfaces, light sources and their interaction? The plenoptic function $L(x, y, z, \theta, \phi, \lambda, t)$ assigns radiance to every point (x, y, z) in space and for every direction (θ, ϕ), depending on the wavelength λ and time t, compare [AB91]. Radiance quantifies the amount of radiation that passes trough an area under a given solid angle and direction.

With static scenes and handling of the spectrum of light using three different spectral bands, as the human visual system cannot discriminate arbitrary wavelengths [WILH11], the plenoptic function is reduced to five dimensions. Under the assumption that the medium in which the scene emitting the light field is placed is completely transparent, the radiance along a ray in this medium becomes constant, and the light field outside of the convex hull of the scene can be expressed with four dimensions [LH96, GGSC96]. Two dimensions for the coordinate on the convex hull and two for the angle of the ray. From now on, the term light field denotes this 4D simplification if not stated otherwise. A perspective camera, as modeled in Section 2.1.2, represents a small 2D extract from the light field. By taking a sufficient number of images, captured from locations on the convex hull, it becomes possible to measure the full 4D light field.

Note that there are different parametrisations for a light field. Common is the representation introduced in the first works on light fields [LH96, GGSC96], using two reference planes and indexing rays by giving the coordinates (u, v) and (s, t) of the intersections of a ray with those reference planes, see Fig. 2.4. This parametrisation simplifies geometric calculations for rendering, leaning itself well to implementations on graphics hardware [LH96]. Note that different parametrisations are trivial to interchange, all basically defining a 3D coordinate and an angle for each light ray, in some form or the other. But, different parametrisation may be useful, depending on the application or argumentation, see Fig. 2.5 for an alternative representation. Indeed throughout most of this work the specific parametrisation will not be relevant, if not mentioned explicitly.

2.2.1 Light Field Capture

Several approaches to light field capture have been introduced over time, see [WILH11] for a more detailed overview. Two main directions can be discerned: The first is based on taking several captures from traditional 2D cameras, using multiple cameras or temporal multiplexing. The second is based on 4D to 2D multiplexing, packing the 4D light field onto a single 2D sensor in a way which allows later retrieval of the original light field signal.

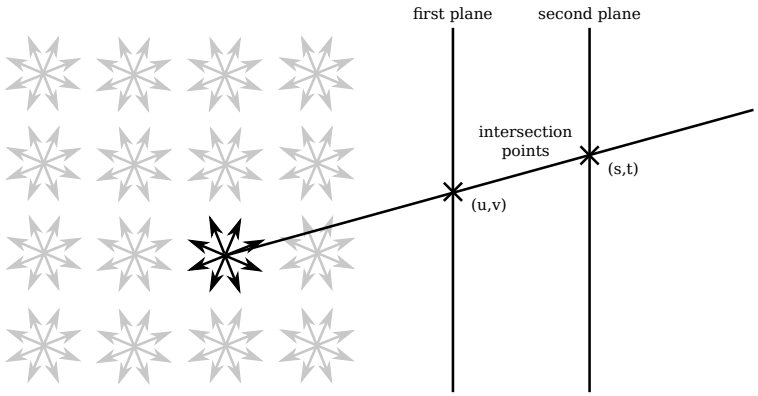


Figure 2.4: Cross view of the light field parametrisation using two reference planes and intersection points.

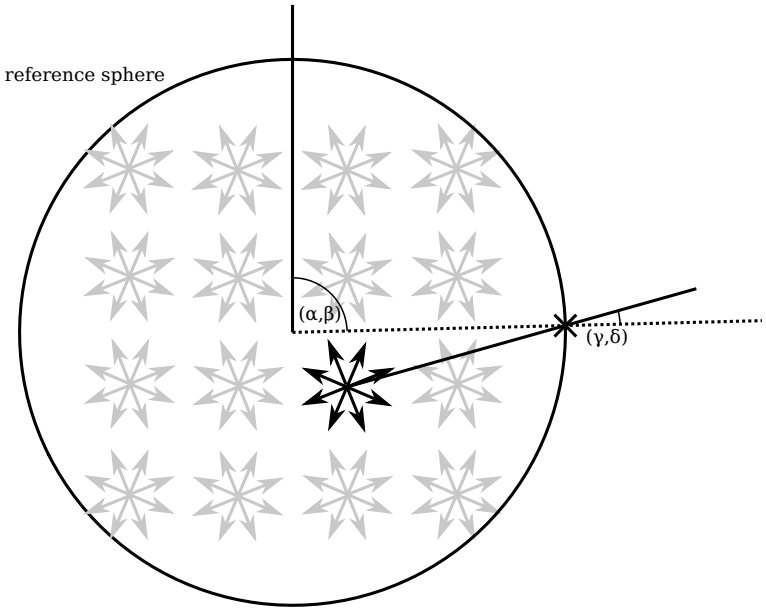


Figure 2.5: Alternative parametrisation, using angles to parametrize the intersection with a reference sphere and the angle to the orthogonal of the intersection point.

2.2.1.1 Using 2D Cameras

Capturing the 4D light field using 2D cameras can be implemented in two ways, either by moving a single camera relative to the scene [LH96, GGSC96], or by using a camera array [LH96, WJV+05], where a number of cameras are placed at the desired positions. The first approach, often implemented using a gantry and possibly a rotary table for full capture from the hemisphere, is simple to implement and can, provided the components are sufficiently

precise, provide basically arbitrary density and resolution of the captured light field. The drawback is that only static light fields may be recorded, as capture is not instantaneous. The second possibility is the use of camera arrays, providing the possibility to capture dynamic light fields and light field video. However this approach is much more expensive to implement, and can provide only limited density, as cameras cannot overlap. The overall number of viewpoints is also more limited as it is more difficult to install additional cameras compared to just taking more images, as in the first approach.

2.2.2 Spatial Multiplexing

Instead of multiple captures with 2D cameras, it is also possible to implement light field capture at a smaller scale by multiplexing the light field reaching the capture device onto a 2D sensor. Different approaches using spatial multiplexing have been demonstrated, based on lenslet arrays [OAHY99, NLMBH05], lenses and prisms [GZC+06] or mirrors [UWH+03, TAV+10]. All these methods basically project small 2D extracts of the 4D light field onto different parts of the sensor, keeping a one on one mapping between light field samples and pixels, which allows easy extraction.

2.2.3 Frequency Multiplexing / Compressive Sensing

A recent development for light field capture is based on the principles of compressive sensing [Don06]. Compressive sensing works under the assumption of sparsity of the input signal, exploiting the large redundancy in light fields. Frequency multiplexing is applied, to encode the 4D light field onto the 2D sensor, and computational methods are used, to retrieve the original signal from the encoded data.

Practical demonstrations of the feasibility of compressive sensing for light field sampling were first demonstrated by Veeraraghavan et al [VRA+07], using binary masked apertures under the assumption of lambertian surfaces. Liang et al. [LLW+08] use a liquid crystal array to capture several images with differing aperture patterns and lift the lambertian requirement, but require several images, one for every viewpoint. Ashok and Neifield [AN10] use an aperture, coded with different levels of opacity, arranged as a rectangular grid, and investigate different bases for aperture coding, together with a linear reconstruction incorporating statistical correlation. Babacan et al. [BAL+12] improve upon the reconstruction using a Bayesian framework and utilize random aperture patterns. Marwah et al. [MWBR13] introduce the notion of 4D light field atoms as the building blocks for light fields. Atoms are derived from a training set of light fields, and used for the reconstruction from coded aperture captures, using optimized coding patterns.

Common to the compressive sensing approaches is the compute intensive reconstruction step and the dependency on the sparsity of the signal. Therefore a light field dataset which contains difficult characteristics should provide a good basis for the evaluation and improvement of such methods.

2.3 Sampling

The models in Section 2.1.2 describe the the projection and the optical effects, which determine the projected scene image reaching the image plane of a camera, without detailing the actual capture of such a signal. Just from the need of a digital representation it is apparent that the signal has to be expressed with a finite number of discrete values. The process of converting the continuous input signal into a finite number of discrete values is called sampling, and sampling theory is concerned with the representation, as well as reconstruction of continuous signals using discrete samples.

2.3.1 Aliasing

Aliasing describes the effect, that different frequencies, sampled with a fixed pattern, may obtain exactly the same sample values. If such aliasing occurs it becomes impossible to unambiguously determine the amplitude of the affected frequencies from the sampled values, compare [Tuc97, Section 39.5].

2.3.2 Sampling of Bandlimited Signals

Consider a function $g(x)$ and its spectrum S(f) obtained with the Fourier transform in Eq. (2.10), compare [Sze10, Chapter 3.4]:

$$S(f) = \int_{-\infty}^{\infty} g(x) e^{-i2\pi f x} \mathrm{d}x \tag{2.10}$$

A function is called bandlimited if the spectrum $S(f)$ has finite support, which means a highest frequency f_l exists: $\exists f_l : \forall f \geq f_l \Rightarrow S(f) = 0$ [Tuc97, Section 39.2]. In this case it is possible to sample the signal with a finite number of samples and without aliasing, which means an unambiguous reconstruction of the input signal is possible. The sampling theorem, introduced in this form by Shannon [Sha48], see also [Tuc97, Section 39.3], gives the minimum rate at which the signal has to be sampled to avoid aliasing, as $2 \cdot f_l$. The rate $2 \cdot f_l$ at which sampling becomes alias free is commonly referred to as the Nyquist rate, while the Nyquist frequency is the maximum frequency which is alias free for a given sampling rate. Hence the Nyquist frequency is equal to f_l when sampling at $2 \cdot f_l$. Light fields however are in general not

bandlimited [ZC03, DMMV05]. To make unambiguous capture of a part of the signal possible, it is therefore necessary to filter out the higher frequencies, effectively restricting the resolution of the signal, a process known as prefiltering, anti-aliasing or application of a bandlimiting or low pass filter [Uns00]. The frequency f_l is also known as the cutoff frequency.

2.3.3 Modulation Transfer Function

The modulation transfer function, short MTF, describes the modulation, respectively attenuation, of a spectrum by a filter. For a lens camera system this can be used to describe the effect of the optics on the spatial spectrum of the image. Specifically Eq. (2.11) gives the modulation spectrum $M(f)$, depending on a frequency f from a known incident spectrum M_i and the effectively observed spectrum M_{eff} on the image sensor, see [Goo96, Chapter 7].

$$M(f) = \frac{M_{eff}(f)}{M_i(f)} \tag{2.11}$$

Note that, in general, analytical derivation of the MTF is difficult, as it requires exact knowledge of the optics and the sensor characteristics, including the microlens configuration. A practical method to derive the MTF from test images is based on the estimation of the point spread function, short PSF, describing the spread of a point light source projected onto the image plane by the optical system. The PSF can by calculated with subpixel accuracy, by observing the edge spread of slanted lines, see [RPN91]. The MTF is then derived using the Fourier transform of the PSF.

2.3.4 Resolution

The resolution of a system describes how well features in a signal can be resolved over a distance, and conversely from which frequencies on the signal experiences a certain amount of attenuation. Common in optics is the definition based on the modulation transfer function, using a cutoff frequency f_c where $\forall f \geq f_c \Rightarrow M(f) \leq t$, normally with a modulation of $t = 0.5$. This frequency is referred to as MTF50, referring to the modulation of 50%. Regarding the aliasing in a discreetly sampled system with a sampling rate equal to $2 \cdot \texttt{MTF50}$, there exist aliasing components, with a maximum magnitude half that of the full signal. For MTF plots of camera resolution, the frequency is often given in cycles per pixel, short C/P, or line pairs per millimeter, short Lp/mm. Note that while a value of $t = 0.5$ is common, other definitions exist. For example the Rayleigh criterion common for diffraction limited signals, see Section 2.1.5.3, states that the peaks of point light sources, filtered by diffraction, can still be distinguished as long as their distance on the image plane is less than the radius to the first zero of the Airy patter [PW13, Chapter 11.2]. But when regarding the MTF of an optical system only limited by diffraction, the equivalent to the Rayleigh criterion is given by the MTF10 value [WB01]. This shows how the notion of resolution dependents on the specific purpose, and that generic resolution estimates are difficult.

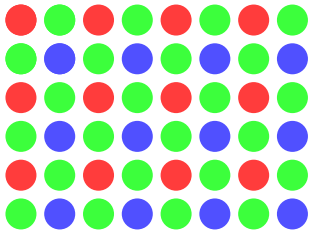


Figure 2.6: An example Bayer pattern. For each sample only one of the three color channels is recorded, using a color filter array in front of the sensor. Different layouts exist, but the depicted distribution is common, providing more green samples, as green obtains the highest sensitivity in the human eye, as well as for common sensor implementations. The pattern repeats every two pixels in both directions, meaning maximum resolution of red and blue is half the full resolution.

2.3.5 Bayer Pattern Sensor

For color image acquisition, the ubiquitous Bayer pattern sensors provide an additional complexity for image sampling [Sze10, Chapter 2.3]. The Bayer pattern allows color image acquisition using a single sensor, by placing a color filter array in front of the sensor, which filters out all but a small spectrum of the light, providing the different color channels. The color filters are distributed over the pixels so that directly neighboring pixels do not record the same color, see Fig. 2.6 for one possible layout. The problem is that now gaps exist in the individual color channels, which have to be interpolated from surrounding values and from the other color channels, a process called demosaicing. The problem with this approach is that the performance depends a lot on the image content. In the best case color information in the image is uniform, and relative brightness at each pixel may be calculated using an adequate interpolation scheme. However in the worst case the color channels are not correlated at all, like for example with contents consisting only of shades of a single color. In this case the sampling rate for red and blue is effectively halved in each dimension. This means that for correct sampling in the worst case, a low pass filter would have to be added to avoid aliasing. However in the best case this low pass filter removes high resolution image contents, which could actually be resolved without aliasing. Indeed, earlier camera designs often included strong low-pass filters to avoid aliasing, while current designs tend to reduce the strength of these filters to gain additional resolution, relying on sophisticated image processing to conceal aliasing artifact. However, strong aliasing is impossible to remove and such designs can easily be provoked to show the respective artifacts. Note that normally the green channel obtains double the number of samples, compared to the red and blue channel, as the human visual system is more sensitive to green light. The takeaway is that correct sampling for arbitrary inputs requires a low-pass filter, to obtain a resolution half that of the full sensor, resulting in a low resolution image.

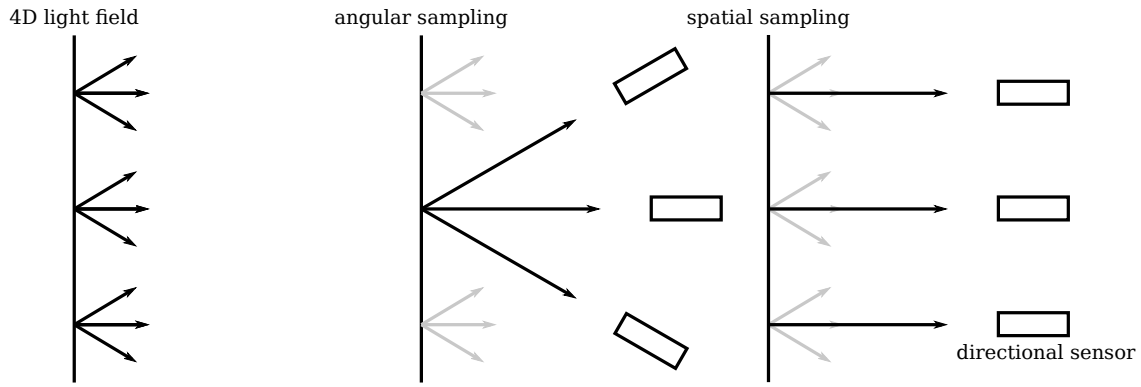


Figure 2.7: Schematic view of the angular and spatial dimensions of a 4D light field, and sampling thereof with a directional sensor, which is sensitive to light in one direction only. Arrows denote light rays from the light field. Note that the surface shown here is just an arbitrary plane outside the convex hull of the field, the light field at other positions can be derived by ray tracing.

2.3.6 Light Field Sampling

The 4D light field signal and sampling thereof, has been examined before. Chai et al. formulate the problem in the Fourier domain [CTCS00] and simplify analysis to lambertian surfaces and non-occluded scenes, which are unlikely to occur in reality. Cha Zhang and Tsuhan Chen [ZC03] expand this work to include occlusion and non-lambertian surfaces and conclude that occlusions cause the spectrum to be bandunlimited. The same conclusion is reached by Do et al. [DMMV05] by investigating textures pasted onto functional surfaces. There are several ways of coping with undersampled light field data, for example using reconstruction methods, which try to reduce artifacts to give plausible results [SYGM03], or, if depth estimation is possible, to increase resolution by exploiting aliasing to achieve superresolution [BZF09]. But those techniques are not free of artifacts. Therefore alias free sampling requires the consideration of bandlimiting effects for correct sampling as detailed in the following sections. The conceptual sampling of the angular and spatial light field components is shown in Fig. 2.7.

2.3.7 Spatial Bandlimit

Fig. 2.8 shows how the spatial resolution of a light field, sampled with a perspective sensor, e.g. a single sample from a 2D camera, is filtered by the resolution of the optical system. This means that the spatial components reaching the sensor can be modeled with the MTF of camera and sensor, to calculate the effective resolution relevant for light field sampling.

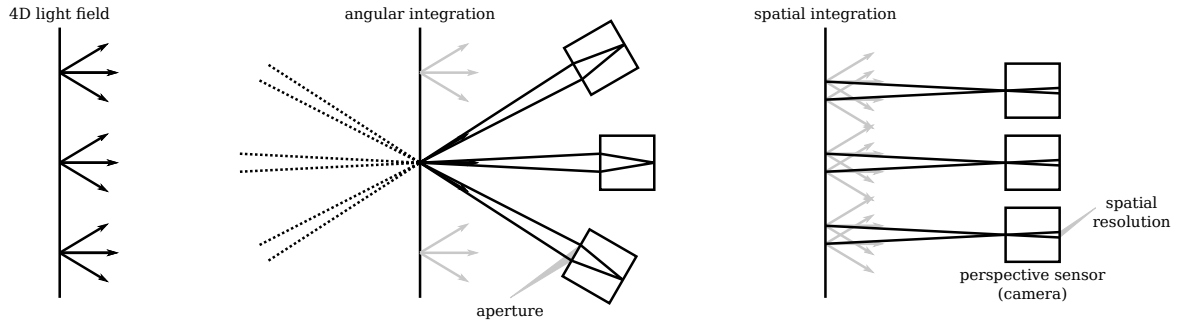


Figure 2.8: View on the bandlimiting elements in an optical system based on a 2D camera. The angular domain is integrated and filtered by the aperture, representing a box filter, see Section 2.3.8, while the spatial domain is filtered with the point spread function of the lens/sensor combination, see Section 2.3.7.

For effective sampling, the sensor resolution translates into a spatial light field resolution by way of the minimal magnification as given by Eq. (2.7). For example given an optical resolution, obtained with the MTF50 on the image plane, of 10 µm, a focal length of 10 mm and a minimal distance from camera to scene of 500 mm, results in a maximum magnification from scene to sensor of:

$$m = \frac{10\,\text{mm}}{500\,\text{mm}} = 0.02$$

The observed spatial resolution of the light field in the scene:

$$\frac{10\,\mu\text{m}}{0.02} = 500\,\mu\text{m}$$

requiring, according to the sampling theorem, see Section 2.3.2, double this frequency for correct sampling, or half the period, and therefore a sampling distance d:

$$d = 0.5 \cdot 500\,\mu\text{m} = 250\,\mu\text{m}$$

Note that the necessary sampling rate is a direct result of the resolution, which itself is a result of all characteristics of the optical system. Specifically the bandlimiting characteristics may be influenced by closing the aperture, to deliberately induce diffraction blur, see Section 2.1.5.3. Another way to artificially lower the required sampling rate, is the use of an optical lowpass filter in front of the image sensor, avoiding the light loss associated with the closing of the aperture, but losing flexibility, as a lowpass filter is difficult to install and exchange. Regarding light fields, the image plane resolution of the 2D camera only limits the spatial components of the light field, which is apparent in Fig. 2.8, the angular components are untouched and may still obtain high frequency content.

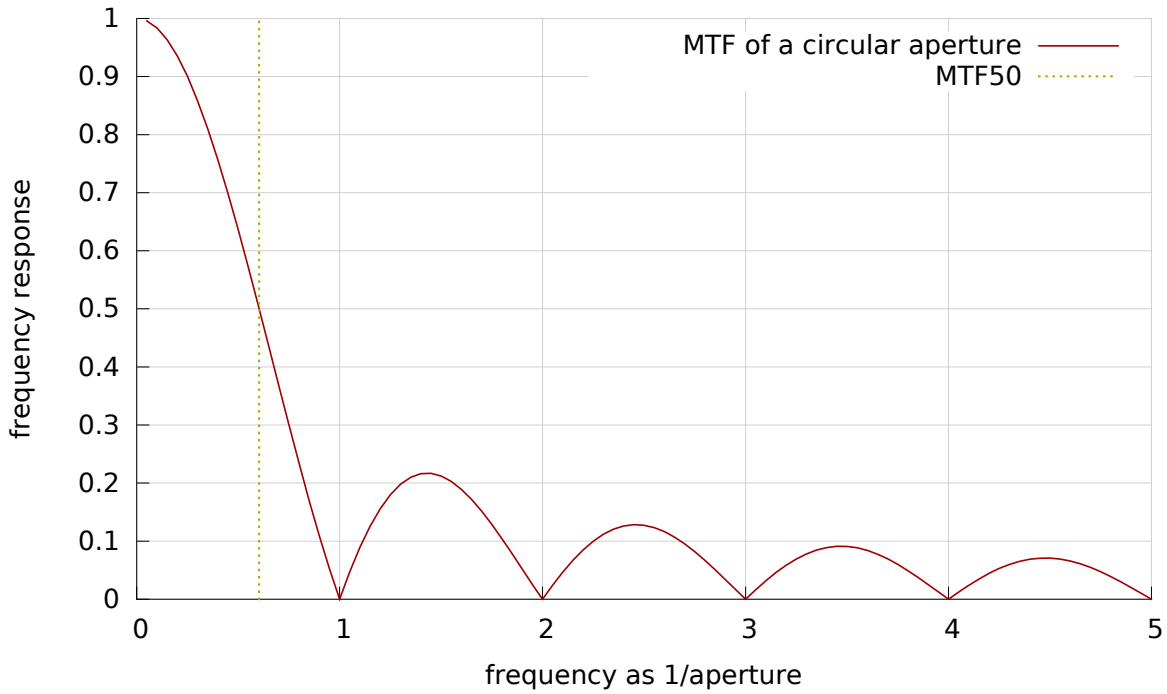


Figure 2.9: Simulated MTF plot for the frequency response of one angular component, using a perfectly circular aperture, represented by a box function. Note that with an MTF50 of $\approx 0.603a^{-1}$ a large amount of the response reaches much higher frequencies, highlighting the bad frequency response of the box filter.

2.3.8 Angular Bandlimit

When regarding the angular components of the light field, the resolution of the recording camera becomes irrelevant. Instead the aperture of the lens is the limiting factor, see Fig. 2.8. However, the lens aperture is normally a box filter, the Fourier transform of which is the sinc function as $\text{sinc}(f) = \sin(\pi f) \cdot (\pi f)^{-1}$, resulting in the MTF plot shown in Fig. 2.9. The MTF50 is obtained by solving $\text{sinc}(f) = 0.5$ with $f \approx 0.603$. Note the unit for f is a^{-1}, where a is the aperture size. Using a small angle approximation, because the aperture is small compared to the focus distance, and for an aperture with a diameter of a, the minimal sampling frequency according to the sampling theorem is therefore double this frequency at $1.207a^{-1}$ respectively the maximum sampling distance d_a is:

$$d_a = 1.207^{-1}a = 0.83a \tag{2.12}$$

A box filter is a poor choice for a low-pass filter, as its frequency response, represented by the sinc function, has infinite support with poor decay, keeping significant frequency content even at frequencies much higher than the cutoff frequency [Uns00]. A few lens designs with non-circular apertures exist, utilizing an apodizing element in the light path to obtain a smoother

aperture in the interest of more pleasing out of focus rendering. For example the Minolta STF 135mm f/2.8 T4.5 and its newer version by Sony [Fot], as well as a similar lens by Fujifilm, the Fujinon XF 56mm F1.2 R APD [Fuj]. However those lenses are not compatible with the requirements of light field capture, being designs with long focal lengths, to easily produce out of focus backgrounds, while light field capture requires a large depth of field.

2.4 Compression

Data compression in general can be divided into two categories, lossless and lossy. As the name implies, the requirement for lossless compression is an exact reconstruction of the original signal on decompression, whereas lossy compression allows a trade off between the compression ratio and the quality of reconstruction. This work focuses on lossless compression, necessary because a reference dataset intended for research and evaluation must not introduce any bias into the dataset. Compression schemes normally consist of multiple steps, transforming and decorrelating the input, often without changing the actual size of the data, but bringing it into a form suitable for the actual compression step. For example video compression often relies on a block based motion compensation to align similar parts in consecutive frames and then stores only the difference between the compensated and the observed value, which can be represented using less bits.

2.4.1 Entropy Coding

Entropy coding uses the uneven distribution of symbol probabilities for compression, by using shorter codes to represent frequent symbols and longer codes for less frequent ones. If unique bit-codes are used for each symbol, then compression can be inefficient, as the probability of occurrence must be rounded to the next full bit. Methods not working on the bit level are able to achieve fractional code lengths, see Section 2.4.1.3 and [Say05, Chapter 3]. Also the modeling of the probability distribution plays a large role and can be derived and transmitted in different ways. The distribution may be fixed beforehand for higher speed or calculated for a range of the input, and transmitted out of band. The distribution can also be derived from the previously observed input stream, avoiding the necessity of transmitting it for decoding and allowing simple adaption to changes in the distribution. Also the distribution may be adapted depending on the context of the symbol to be coded, allowing different models to be used, based on various properties of the input stream.

2.4.1.1 Universal Codes

Instead of individual symbol probabilities, universal codes encode integers under the assumption of some monotonically decreasing distribution, assigning shorter sequences to smaller numbers [Mac02]. Exact knowledge of the probability distribution is not required and methods are therefore well suited to compress large integers, where it may not be possible to derive probabilities from prior observed values, because of the large code space. Examples include Golomb, Rice and Elias gamma and delta coding, which are reviewed in the context of fast integer compression in [LB12]. The common idea is to code integers only with the number of significant bits, stripping leading zeros, with the specific coding scheme being the main difference between different methods. Inefficiencies arise, because the number of significant bits also has to be transmitted for each symbol. On the other hand universal coding schemes are very simple, making them suitable for fast implementations.

2.4.1.2 Huffman Codes

A Huffman code is an optimal, variable length code with a fixed mapping, which maps each input symbol to a code word [Say05]. Compression is achieved by varying the code word length so that high frequency symbols are represented by shorter codes. The code represents a prefix code, which means no concatenation of valid codes may result in the prefix of a valid code, else decoding would be ambiguous. Prefix codes are also known under the term prefix free code and instantaneous code. Huffman coding is simple, and encoding and decoding can be implemented using a simple lookup table which leads to fast encoding and decoding speeds. However, symbols have to be handled one at a time, which limits the performance as it is difficult to exploit SIMD extensions of modern CPUs. SIMD is short for single instruction multiple data, a concept which allows a single instructions to process multiple elements in parallel, multiplying performance if applicable. Also the mapping has to be constructed at one point, and explicitly updated to adapt to changing probabilities, which is expensive. Code words cannot have a rational length, which leads to inefficiencies [Mac02], as code words cannot be shorter than one bit and compression of an 8 bit input alphabet can never exceed a ratio of 1:8.

2.4.1.3 Arithmetic and Range Coding

Instead of mapping an input symbol to a fixed code word, arithmetic coding represents the whole input sequence as a single number [Say05]. The coder starts with a lower and upper bound, which for every input symbol is restricted to a range representative of the probability of the respective symbol. The sequence is uniquely represented by this range. Compression is achieved because symbols with high probabilities result in small range restrictions, allowing the final range to be represented with a shorter number.

For example consider a two symbol alphabet, using the letters 'a' and 'b' and probabilities $p(a) = 0.75$ and $p(b) = 0.25$ the sequence 'aab' is then encoded with the ranges, letter by letter, starting with the initial range: $[0,1]$, $[0.75,1]$, $[0.9375,1]$, $[0.9375,0.953125]$. On decompression the decoder can determine the respective symbols by looking at the range for each symbol. From the final range it is clear the first letter has to be 'a' because the output of the encoder lies between 0.75 and 1, while for 'b' it would have to be between 0 and 0.25. Note that an actual implementation has to deal with the limited precision and range of the used number system, and has to periodically flush and normalize the range.

Arithmetic and range coding only differ in the range representation. Range coding uses integers, arithmetic coding relies on a rational representation of limited precision. Adaptability is trivial to add, as probabilities can be changed at every coding step, but performance is much slower than using Huffman codes at an increased efficiency, because probabilities do not need to be rounded to a binary code.

2.4.1.4 Asymmetric Numeral Systems

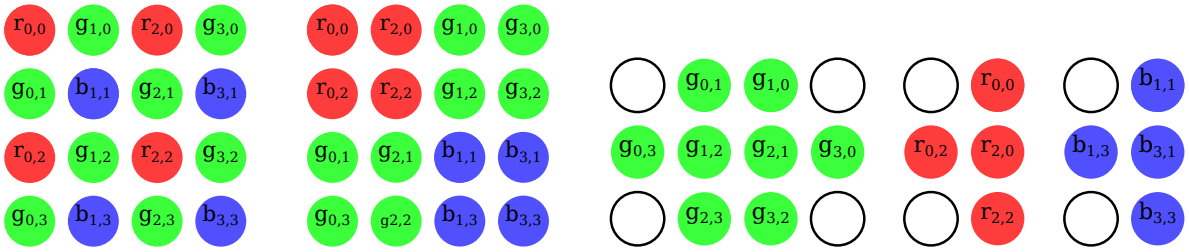
Asymmetric numeral systems, introduced by Duda [Dud09] are an abstraction to range coding as used above, enabling a range of optimizations like simpler state representation and table based implementations. Basically ANS allows to combine the compression performance of arithmetic and range coding, with the speed of a Huffman coder.

2.4.2 Decorrelation

Moving away from individual symbol probabilities, towards the modeling of the input stream: If correlation exists within the input stream, then this correlation may be used to derive a decorrelated representation which is more suited for entropy coding [Say05]. As an example, consider an image consisting only of a single linear gradient, slowly changing from completely black to white. While the absolute pixel values are evenly distributed, giving no leverage to entropy coding, the difference between adjacent pixels consists exclusively of small values, a very uneven distribution easily compressible by entropy coding. Decorrelation depends heavily on the use of a good model, which captures as many specifics of the data as possible.

2.4.2.1 Color Space

Regarding the representation of color images, the common representation on the basis of the primaries red, green and blue is suboptimal for compression, as correlation is high. Often when one primary changes the others change accordingly, i.e. most information is contained in the brightness component, while color is more constant. Therefore for the compression of color images, a conversion from RGB into an other color space is useful, providing a different



(a) raw Bayer **(b)** side by side **(c)** rotation scheme

Figure 2.10: Visualization of the different methods which pack a Bayer pattern image into common pixel formats with color subsampling. Fig. 2.10a shows the raw Bayer pattern, which can be stored as a grayscale image. Fig. 2.10b demonstrates the packing as a rearrangement of color samples. In Fig. 2.10c the rotation scheme is visualized, where the image is rotated by 45° to align it onto a regular grid, with two color channels at half resolution.

representation of individual pixel values using different primaries. Common are the YUV variants, where Y is the brightness, a weighted sum of red, green and blue, and U and V are differences between the color channels. Note that the main application for such color spaces is lossy compression, therefore the non-linearity of human perception is normally incorporated in such models, making lossless transformations difficult. An example for a lossless color space is the reversible color transform, used in `jpeg2000` [CSE00], which however increases bit depth to 9 bits for two of the channels.

2.4.2.2 Bayer Pattern

An additional difficulty arises for the lossless compression of Bayer sensor images. A Bayer sensor samples color at one channel per pixel, see Section 2.3.5, a format which is not supported by common compression methods. The normal approach for lossy compression is the demosaicing of the Bayer pattern, a step which interpolates the missing color information for all pixels in the Bayer pattern. This approach is not useful for lossless compression as it increases the amount of image data by a factor of 3. Three methods to provide Bayer pattern images to a not Bayer aware image compression schemes are introduced in the following. Fig. 2.10 visualizes the different approaches.

Raw Pattern: The most basic way to represent Bayer patterns is as a single channel gray scale image. This is sub optimal for most compression methods, because there is high fluctuation between the interleaved color channels, but notably `jpeg2000` performs quite well, see Section 3.2, an effect of the wavelet decomposition, compare [ZW06].

Side by Side: While most methods do not provide four channel image formats, the Bayer pattern may be decomposed into the individual channels, placed side by side in a single grayscale image. This means the final image is split into four quadrants, with each quadrant representing a single color channel as a scaled down image, see Fig. 2.10b. This approach can provide better performance for most compression methods due to the better spatial correlation.

Packing by rotation: The third method exploits color subsampling, present for most pixel formats used in image compression. Color subsampling describes a lossy compression step where for an YUV-style color space the color channels U and V are stored at a reduced resolution, because the human visual system is more sensitive to detail in the brightness channel than in the color channels. This does allow the packing of a Bayer pattern, as the green channel obtains double the resolution compared to red and blue. Zhang et al. [ZW06] examined different methods of placing the Bayer pattern in a way which maximizes compression, and found that a rotation scheme provides the best performance, see Fig. 2.10c for a visualization. The image is rotated by 45 degrees, which allows the alignment of the green channel to a rectangular grid. The blue and red channels are treated the same, only at half the resolution, providing the color subsampling. As the image is rotated, missing samples have to filled in towards the corners, for example with a constant color, which compresses very efficiently for most methods. Also the red and blue channel are not stored directly but as their difference to green, providing the color space decorrelation.

2.4.2.3 Prediction

If the correlation model allows prediction of the next symbol from already decoded symbols with some accuracy, then this prediction can be used for compression, by storing the difference between prediction and observed value. For images, prediction is normally based on neighboring pixels, trying to detect and continue edges and gradients.

2.4.2.4 Disparity/Motion Compensation

In video or light field data, groups of pixels often move in similar direction if they belong to the same object, which is exploited by motion compensation, in video compression, or disparity compensation, in stereo, multi-view and light field data. If a block only changes location between different images, it is possible to code the offset plus the residual after compensating this offset to efficiently describe the object over several frames. This principle is applied to video, multi-view video and light field compression, with the latter giving even more references in more dimensions than the former. Strictly speaking this is a special form of prediction, although very distinct to local, texture-based prediction.

3 State of the Art

This chapter provides an overview and qualitative evaluation of existing works. Three subjects are examined, first the main topic of this work which is the capture of a dense light field dataset. Therefore publicly available light field datasets are evaluated. Secondly, related work in the scope of fast lossless data compression is introduced and compared, including a benchmark run on the light field dataset recorded in this work. Thirdly, the current state of the art on fiducial markers, used for accurate camera calibration, is briefly introduced and evaluated.

3.1 Existing Light Field Datasets

Six light field databases are known to the author, which are available to the public, including synthetic and real captures.

- The (Old) Stanford Light Field Archive
  http://graphics.stanford.edu/software/lightpack/lifs.html
  Contains mostly computer generated light fields, with the highest number of viewpoints of any of the datasets at up to 64×64 viewpoints. However, resolution is extremely low, images have a maximum resolution of 256×256 pixels.
- The (New) Stanford Light Field Archive
  http://lightfield.stanford.edu/lfs.html
  Collection of a number of light field recordings obtained using gantries [LPC+00], a camera array [WJV+05] and a light field microscope [LNA+06]. Individual images reach up to 1536×1280 pixels. Viewpoint resolution is relatively low, light fields are mostly 17x17 views and the viewpoint spacing varies.
- Synthetic Light Field Archive
  http://web.media.mit.edu/~gordonw/SyntheticLightFields/
  A set of synthetic light field renderings. Viewpoints are relatively dense at a spacing of $\leq 1\,\text{mm}$, however the number of viewpoints is quite limited at mostly 5×5 views [MWBR13].

- Middlebury Stereo Datasets
  http://vision.middlebury.edu/stereo/
  Recorded images meant for stereo correspondence, which explains the rather large baselines and the low number of up to 7 viewpoints. Includes ground truth for depth, gathered using structured light. [SHK+14]
- Datasets and Benchmarks for Densely Sampled 4D Light Fields
  http://hci.iwr.uni-heidelberg.de/HCI/Research/LightField/lf_benchmark.php
  Datasets for the evaluation of disparity and segmentation in light fields, containing recorded and rendered images together with depth and segmentation ground truth. While baselines are smaller than earlier datasets the number of views is also limited at 9×9 [WMG13, Wan14].
- Disney Research
  http://www.disneyresearch.com/project/lightfields/
  Several datasets of up to 151 very high resolution images with a baseline down to 2 mm, provided as part of a work on high resolution scene reconstruction [KZP+13]. The datasets represent only 3D light fields with viewpoints placed along a line.

At the time of writing the UCSD/MERL Light Field Repository referenced by [Wan14] was not accessible.

Additional datasets exist, which are not available to the public. The largest the author is aware of is the recording of Michelangelo's statue of Night by Levoy et al. [LS99] with 24304 viewpoints and a viewpoint spacing of 12.5 mm. While this may be the largest light field dataset to date, the viewpoint spacing is still rather large and lossy jpeg compression was used to reduce storage requirements, making the dataset less suitable for evaluation and reference work.

3.1.1 Sampling

While some of the datasets mentioned in Section 3.1 contain quite small baselines ≤ 1 mm, it is difficult to evaluate if this represents sufficient sampling. Scene geometry and the resolution of the optical system plays are large role, see Section 2.3. However for the datasets mentioned in Section 3.1 no documentation exists towards the consideration of sufficient sampling. Some of the datasets may be densely sampled, but it is not clear how the respective sampling rates were determined. Additionally most available datasets place viewpoints within a relatively small space, which is sufficient to produce synthetic aperture renderings, but limits camera movement to this small range. Therefore a large dataset, for which sufficient sampling was estimated and implemented would be highly beneficial.

3.1.2 Conclusion

In summary, there does not yet exist a light field dataset which provides both dense sampling and a large viewpoint area. Additionally, no direct reference for sufficient sampling is available for any of the datasets. This work tries to give an answer to the question of dense sampling, providing an estimate of sufficient sampling, see Section 2.3, and a dataset sampled according to this estimate.

3.2 Fast Lossless Compression

The following will give a brief evaluation of lossless compression methods with a focus on high speed, sorted by dimensionality. The dataset recorded in this work is intended for evaluation and reference work, therefore lossy compression, which would introduce an unknown bias into the dataset, is not acceptable. Also compression needs to be fast, or it will become the bottleneck of the whole system. Compression requires a bandwidth of least 360 MiB s^{-1} and decompression up to 1600 MiB s^{-1}, see Section 4.5.

The capture system used in this work provides the light field data as a video stream which has to be compressed on the fly. The light field nature of this video stream is not directly available to the compression methods, as a 3D slice consisting of more than 13000 images has to be processed before the next slice may be recorded, and buffering of such a large amount of data prior to compression is not practical. However, no lossless light field compression method exists which could make use of the fourth dimension, see Section 3.2.4 below, so this fact has no impact on the evaluation. The evaluation is sorted by dimensionality, denoting the number of dimensions considered by the compression method. Methods with lower dimensionality may be used to compress data with a higher one, tough at the expense of compression efficiency, as correlation in the dimensions not considered cannot be exploited. On the other hand, lower dimensionality can lead to a reduced computational burden, at least low dimensional methods tend to achieve faster speeds. Fig. 3.1 shows results for most methods mentioned, allowing comparison of the achieved bandwidth of the respective methods, as well as evaluation of the trade off between speed and compression ratio.

3.2.1 Text and 1D Compression

Generic 1D compressors like `deflate` (`zlib/gzip`) [Deu96], `bzip2` [Sew96] or `lzma` (`7z/xz`) [Pav] are optimized for high compression and hence do not reach high speeds, but reach quite good compression ratios, especially considering that they cannot directly exploit the high dimensionality of the light field data. However, a number of very fast compression methods have been introduced over the years, most of which are closely based on the original `lz77` compression scheme [ZL77], and trade off compression efficiency for higher speed. Those

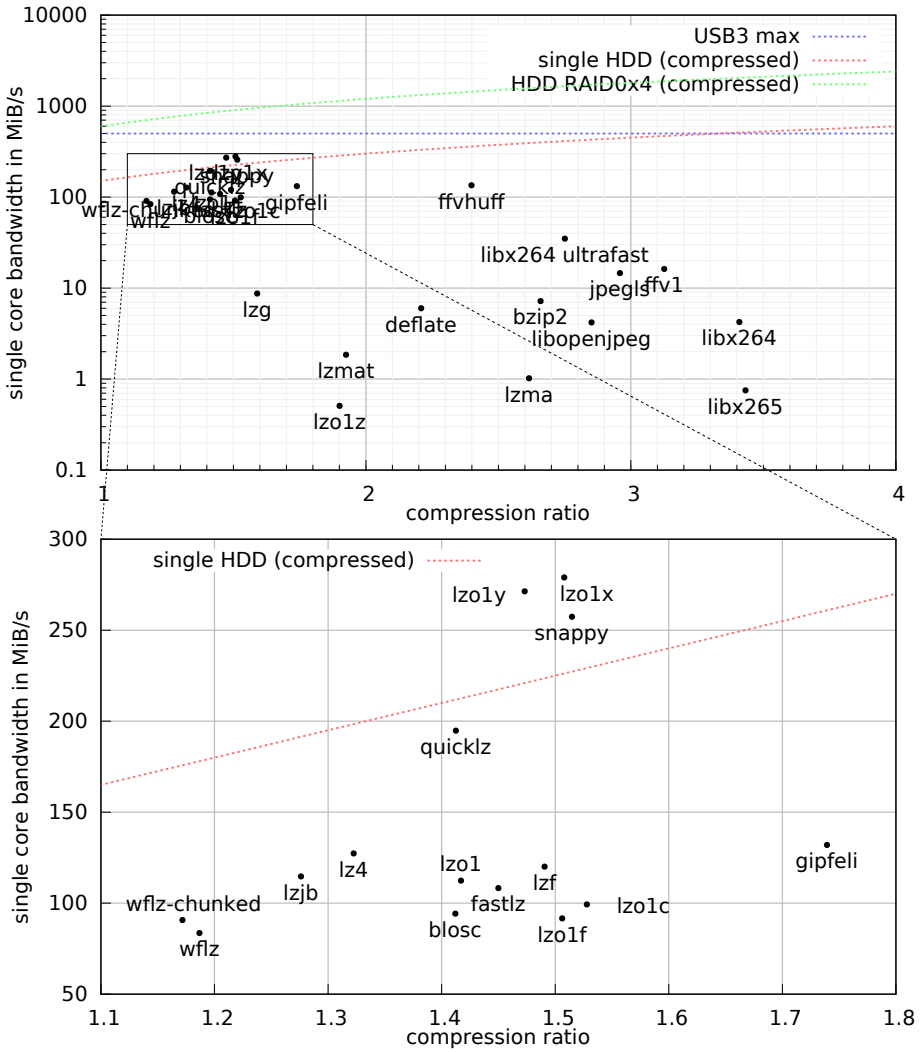


Figure 3.1: A range of lossless compression methods. The upper plot shows all tested methods. The fast generic compression methods obtain similar speeds and are depicted again on the lower plot for clarity. Note the logarithmic scale for the vertical axis in the upper plot. All results were obtained on a single core of an Intel® Core™ i7-860 Processor at 2.8 GHz. For the most part the generic methods are dominated by the specialized image and video compression methods. Only for the highest speeds there are no alternatives to the fastest generic methods like `lzo1x`, which however obtain a poor compression ratio. The first plot also includes the maximum bandwidth of USB3 for an estimation of possibly required compression bandwidth, and an average HDD in single, as well as a 4× RAID0 configuration, to estimate required output bandwidth. The HDD plots include the respective compression ratio, simulating the effectively available bandwidth.

methods are used in various projects from filesystems to databases and include: `lzo` [Obe], `snappy` [SNA], `fastlz` [Hid11], `quicklz` [Rei11], `gipfeli` [LA12] and `lz4` [Col11]. For the dataset evaluated in this work, the fastest was `lzo` at a speed of 280 MiB s^{-1}. See Fig. 3.1 for an overview of the respective performances. This is consistent with the results achieved by Gomez et al. in [GCAJ+13], who evaluate those coders for 4K video compression. The disadvantage for these fast methods is the very limited compression ratio as dictionary based methods are not well suited for the compression of image data. Even faster speeds are achieved by specialized methods utilizing bitpacking to compress 32 bit integers [LB12] or 64 bit floating point values [BR09]. Specifically the method by Lemire and Boytsov [LB12] incorporates a SIMD implementation to reach a speed in excess of 8 GiB s^{-1} on a single core. However, as the image data recorded for a light field image consists of 8 bit values, the method is not directly applicable for light field compression.

3.2.2 Image Compression

Dedicated lossless image compression methods like `jpeg-ls` [WSS00] peak at around 25 MiB s^{-1}, with modifications reaching up to 75 MiB s^{-1} [WKG+12] on an Intel® Core™ i7-920 Processor at 2.67GHz. The `jpeg-ls` standard is based on the LOCO-I algorithm [WSS00], was standardized in 1999 and is still widely used as baseline for the evaluation of lossless image compression methods. Despite the gap between these methods and the fastest 1D compression methods, amounting to two orders of magnitude, there do not seem to exist any faster image compression methods. Newer developments concentrate on increased compression efficiency at the expense of compression time, which is not the focus for this work.

3.2.3 Video Compression

Similar to image compression, research seems less focused on fast lossless video compression, when comparing with lossy compression. However, video encoders implementing the state of the art `HVEC` standard [X2617], or its predecessor `AVC` [X26], include lossless profiles which provide high compression ratios. Also a number of open source implementations of simple lossless video compression methods are available in the `ffmpeg` library [FFM]. Notably `ffvhuff`, an adaption of `HuffYUV` [Tog03], combines a simple Huffman coder with the median predictor from `jpeg-ls` and achieves the fastest compression speed at 132 MiB s^{-1}, see Fig. 3.1. On the other hand, `ffv1`[Nie13] uses the same predictor with a context adaptive range coder, similar to the arithmetic coder from `AVC`, and achieves a performance trade off competitive to `AVC`.

3.2.4 Lossless Light Field Compression

The author of this work is not aware of any dedicated lossless compression scheme for light field data. However for HVEC, the successor of AVC, extensions have been demonstrated which provide increased compression efficiency for light field and multi view sources, by exploiting the additional redundancy [CNS12, CKB+14, LSOJ14]. No evaluation of lossless performance has been performed, but those approaches may be seen as an indication that highly efficient lossless coding for light field data could be available as an extension in future standards.

3.2.5 Conclusion

In summary, while generic text compression methods can achieve higher speeds than the fastest video compression methods, this comes at the cost of a significantly reduced compression ratio. Also none of the benchmarked methods reach the speed of the specialized bitpacking schemes. But those are not directly applicable to image based data, which is also the reason they were not benchmarked for this evaluation. A faster compression method for light field data may therefore use the principles introduced by Lemire and Boytsov [LB12] and apply those to light field compression. Indeed this is the method implemented for this work, see Section 5.5.

3.3 Fiducial Markers

For the calibration of the light field dataset, self identifying markers, also called fiducial markers or fiducials provide a simple method of obtaining a number of reference points for calibration. Gortler et al. [GGSC96] already use a circular marker system for the calibration of their hand-held capture system. Such markers have a structure easily recognizable by computer vision methods, and provide a number of identifiable markers to provide more than one reference point. Specifically the method of Zhang [Zha00] only requires a planar calibration target for extrinsic and intrinsic calibration, therefore markers can easily be deployed using a black and white printer and a planar surface, to provide very accurate calibration targets at low cost.

Several fiducial marker systems have previously been proposed. Garrido et al. [GJSMCMJ14] give a good overview over such systems, and conclude that square markers provide more reference points, as a single marker provides four corners, while marker systems based on a circular layout [KGS98, SBGD07] only provide a single reference point per marker. For example ARToolKit [KB99], one of the first of such systems, allows arbitrary patterns to be used for identification, which have to be registered with the toolkit. However, this approach is problematic because the system cannot guarantee the discernability of markers, therefore requiring large markers for good performance. Later approaches use square binary patterns to identify markers, leading to a more predictable performance, as the atoms leading to identification have now the same size and form. The marker systems using square binary

patterns for identification then differ mainly in how detection is performed, and in the way the id is encoded into the binary pattern. Below follows a brief history and evaluation of such systems.

3.3.1 Basic Detection

For all systems, the detection process will be described roughly split into three stages: Normalization, candidate selection and verification. After the detection process the marker is identified using the pattern contained in the center. The normalization step is normally some kind of adaptive thresholding, to allow classification of image contents into the black and white of the binary pattern. Then follows candidate selection, which uses some property of markers, like edges or connected components as a starting point for marker detection, followed by a verification step which tries to match and verify candidates using some intrinsic feature of the marker. After those three steps identification takes care of the retrieval of the id from a successfully recognized marker.

For `ARToolKit` [KB99] the normalization step consists of a fixed thresholding, followed by a connected components analysis for candidate selection. Four lines are fitted to the contour of a candidate blob. The intersections of those lines provide the corners of the ma,rker, which is then reprojected and identified using pattern matching. The fixed thresholding is problematic under all but very controlled illumination and the arbitrary identification pattern may be difficult to identify. Those problems inspired the development of `ARTag` and `ARToolKitPlus`, see below.

3.3.2 Error Correction

Most marker systems based on binary patterns use error correction schemes to improve robustness. The following methods incorporate error correction schemes which can decode the correct id for individual markers, even in the presence of a limited number of errors. They use a scheme which encodes an id with more bits than necessary, using the additional bits for error correction as well as for rotation invariance, as square markers may be detected in any of four different orientations.

`ARTag` [Fia10] uses an edge based approach for normalization. Detected edges are used as candidates which are then grouped into quadrangles. This grouping serves also as the verification step. `ARTag` improves the identification from `ARToolKit` by using an 6×6 grid of black and white squares, coding a 36 bit word which itself includes the 10 bit id plus 26 bits for error correction. Because the marker may be rotated in any direction the error detection also serves to correctly identify the rotation of the marker. However, as Garrido et. al point out [GJSMCMJ14], the number of bit-errors correctable with the used error correction scheme is only 2.

ARToolKitPlus [WS07] improves on ARToolKit by incorporating a 36 bit binary pattern similar to ARTag, with effectively 12 bits usable for marker identification, and with the same robustness as ARTag with maximal 2 bit errors. However the adaptive thresholding uses a single threshold for the whole image, making it unsuitable for multi-marker detection.

Later systems use variations of the techniques introduced above and concentrate on improving robustness by using better error correction codes. Olson [Ols11] constructs near optimal codes for AprilTag using lexicodes, with a maximum of 10 bit errors for 2221 distinct markers and a maximum of 9 bit errors for 4146 markers, both with 36 bit codes. The aruco library [GJSMCMJ14] follows similar lines, using a 25 bit code with 1024 unique markers and a maximum of 2 bit errors.

3.3.3 Accuracy

Concerning the quality of detected reference points, Atcheson et al. demonstrate with their CALTag markers [AHH10], that a corner locator based on saddle-point refinement provides improved accuracy of corner location, compared to line intersection methods. However this comes at a cost, as the saddle refinement needs an area where only saddle point edges are visible, resulting in either a large border for the marker, or extra saddle points, located at specific locations relative to the markers. The used marker system uses a 16 bit error correction scheme for a total of 280 markers with a maximum correction of 3 bit errors.

3.3.4 Reference Point Density

Maximizing the accuracy of camera calibration benefits from a high number of reference points and from accurate corner location. However incorporating error correction, as used in most marker systems, increases the marker size and therefore decreases the density at which markers can be placed. Olson shows for AprilTag [Ols11, Fig. 10], that a good error correction scheme works only up to a certain size. Smaller markers cannot be detected because it becomes impossible to distinguish the individual bits of the bit pattern. Fig. 3.2 shows similar findings using the aruco library. This means that for a fixed number of available markers, error codes actually decrease the density with which markers may be packed, as the identification code increases in size, and with it the smallest size at which markers can be identified. Additionally, markers cannot be placed directly side by side, as most systems would not be able to detect where one marker ends and the next begins, also limiting density. Atcheson et al. [AHH10] address this problem, by using a checkerboard like pattern, where every second row of markers is placed at an offset of one marker. However, their approach severely limits the number of markers and increases corner thickness due to the requirements of the saddle point refinement.

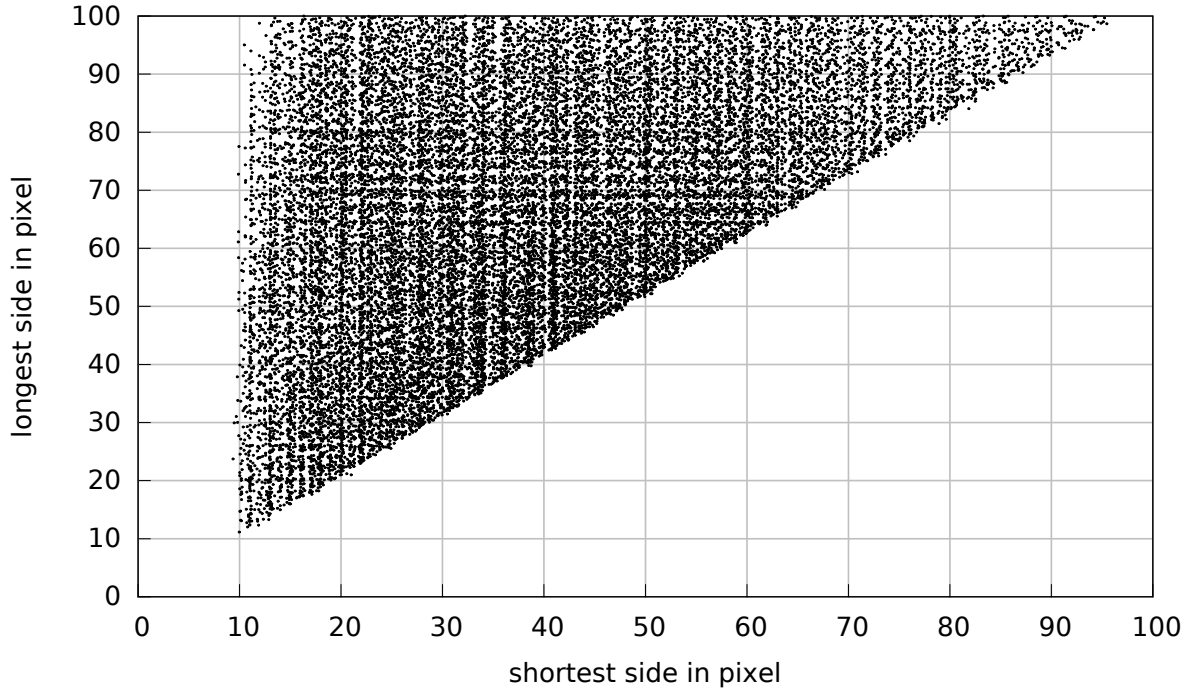


Figure 3.2: Evaluation of the `aruco` marker system for a test image at various sizes. The plot shows the length of the smallest side of detected markers, against the longest side, for successfully detected markers. A single test image was scaled all the way down to a marker size of 1 pixel, for all combinations of horizontal and vertical scale down. The plot shows that `aruco` markers with a size smaller than 10 pixels cannot be identified, despite error correction. Marker size is, independent of error correction, limited by a lower bound which is determined by the number of elements in the marker. As error correction increases the number of elements necessary for the same number of distinct markers, error correction therefore increases the size necessary for successful marker detection.

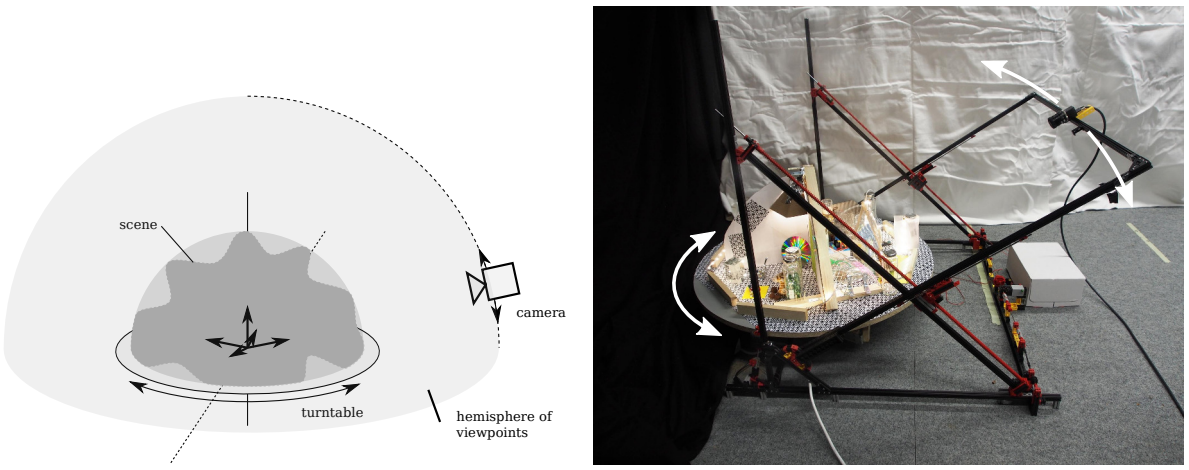
3.3.5 Conclusion

In conclusion, existing marker systems provide only a limited number of distinct markers, while using a larger area for error correction and orientation fixation. Additionally, improved corner accuracy is possible but requires additional space. A marker system maximizing reference point density therefore would have to overcome a number of obstacles:

Error correction: Provide robustness and error correction without increasing marker size, or decreasing the number of unique markers.
Marker count: Keeping the marker size small for dense packing, while still providing a large number of unique markers.
Orientation fixation: Provide orientation fixation without additional space usage.
Corner refinement: Incorporate corner refinement without increasing the marker size.

4 System Design

This chapter will give the detailed reasoning for the respective design choices, according to the modeled properties of the light field and optics from Chapter 2, and detail the relationships and resultant constraints. The chapter starts with an overview of the design, stating the core parameters. The detailed reasoning for this design then begins with the dimensions and the geometry of the light in Section 4.2, going over the mechanical recording setup in Section 4.3, and an analysis of the optical setup in Section 4.4, to the processing and storage requirements in Section 4.5. Finally the required compression is examined in Section 4.6, finishing with the fiducial markers system required for viewpoint calibration in Section 4.7. Technical details and the particularities of the implementation are covered in Chapter 5. Fig. 4.1 shows a schematic and an image of the mechanical part of the setup, highlighting how the light field sampling is realized.



(a) Schematic of the mechanical setup.

(b) Picture of the realized hardware.

Figure 4.1: Two pictures of the hardware setup. Fig. 4.1a shows a schematic of the setup, with dashed lines indicating the vertical axis and the arc of the camera movement. The horizontal rotation is provided by a turntable. Fig. 4.1b shows the actual setup, with the two degrees of freedom indicated with white arrows. The possible viewpoint positions lie on a hemisphere centered at the intersection of the two rotation axis.

4.1 Overview

This section serves to give a brief overview of the system design for dense light field capture, as implemented in this work, as well as core parameters. The detailed explanation for the core parameters and respective design decisions follows below, starting with Section 4.2.

Camera: Dense light field capture requires the capture of large amount of images. To keep capture times down the setup was based around a XIMEA MQ042CG-CM camera which provides a resolution of 2048×2048 pixels, at a rate of up to 90 fps, for a total bandwidth of $360\,\mathrm{MiB\,s^{-1}}$, see Section 4.4.1 for more details. This allows significantly faster capture times than a setup based on single image capture.

Mechanical Setup: A continuous stream of images places special constraints on the mechanical setup, as the target viewpoints have to be traversed at a constant rate. The mechanical setup therefore utilizes a rotary table, turning at a constant rate, which simplifies the sampling of horizontal viewpoints. With the camera fixed in one position, a single rotation of the turntable represents a single horizontal 3D slice of the light field. To achieve viewpoint sampling in the vertical direction, the camera is attached to a motorized arm with a length of 100 cm, which allows up/down rotation. After each full rotation of the turntable the arm is moved one step, allowing sampling of the next slice of the light field. Fig. 4.1b depicts the setup and Section 4.3 gives a more detailed description.

Sampling: The required sampling rates are derived from the measured band limiting properties of the optical system at the given scene geometry. See Section 2.3 for sampling and Section 4.4 for the optical setup. The actual sampling rates are calculated in Section 5.1. The properties of the optical system were chosen to allow sufficient depth of field for single image capture of the whole scene, within the limits of the mechanical setup, see Section 4.4. The captured scene has an diameter of 70 cm and the used lens has a focal length of 12.5 mm for a horizontal and vertical field of view of 48.5°, and 65.0° in the diagonal. With the remaining parameters of the setup this results in a required maximum viewpoint spacing of 420 µm. From the required sampling rate the rotation speed of the turntable can be calculated as 166 s for one revolution.

Storage and processing: Computation and storage was provided by a single consumer grade personal computer, equipped with six 3TB HDDs in a software RAID6 configuration for increased I/O bandwidth and high reliability. The minimal continuous write rate at the inner tracks of the drives was measured at $367\,\mathrm{MiB\,s^{-1}}$. This does not leave a lot of headroom for filesystem and metadata overhead, compared to the constant input data rate from the camera of $360\,\mathrm{MiB\,s^{-1}}$. But the compression, applied to the captured light field primarily to reduce the required storage capacity, also serves to more then halve the required datarate with a compression ratio of more than 2:1, eliminating I/O bandwidth as a bottleneck in the capture process, see Section 6.1. For this reason RAID parameters were optimized to provide maximum read performance for later light field processing, see Section 4.5.

Calibration: Accurate calibration information is crucial, to map images and pixels to the individual ray bundles of the light field. However the mechanical setup does not provide accurate position information and synchronization between camera and control of the robot would be difficult. Therefore a calibration method based on marker patterns placed in the light field is used to provide precise calibration for each captured viewpoint. Because external measurements for accuracy were not available it is difficult to assess the absolute performance of the marker based calibration system. However marker positions are refined to subpixel accuracy and plots of the final calibration results suggest a precision of at least 100 µm, see Section 6.2.2.

Software: The software for recording and processing of light field data has to cope with the large amount of data of the light field. The implementation is capable utilizing the full read bandwidth of the RAID6 array, by issuing a large amount of parallel read requests in background and the capture software eliminates any possible I/O delay by processing writes completely asynchronous, see Section 5.7.1.

4.2 Dimensions

One of the first considerations goes towards the the subject or scene to be captured. Because of the large amount of data capture is not instant, so the scene has to be static. Also scene size and geometry plays an important role. The 4D parametrisation of light fields requires capture to take place outside the convex hull of a scene. If a full sampling of the light field of a scene is required then it is necessary to cover all points on the convex hull. For this reason it is mandatory to keep the convex hull as small as possible and hence also the scene. On the other hand, a larger scene is more representative for real world light fields, and allows different objects and surfaces to be placed in a single capture, giving more complex interactions of reflection and occlusion and therefore a more complex light field, exhibiting more of the irregular structure which may be encountered in a real world light field. While a complete recording of the scene is not a requirement for this work, it allows more throughout exploration of the scene when rendering images and imposes less limits for the rendering of novel viewpoints and is therefore one of the targets of the design.

In summary, the scene should be as large as is practical for recording and dense sampling, and scene size is therefore limited by the mechanical and optical characteristics of the setup, detailed below in Section 4.3 and Section 4.4.

4.3 Mechanical Setup

To capture light fields using a single 2D camera, most approaches utilize a gantry to cover a part of a plane with viewpoints [LPC+00, WMG13]. This approach however does not allow the recording of the full convex hull without manually moving the gantry [LH96, LS99]. Whatever the scene geometry, a sphere of sufficient size can be placed outside the convex hull of the scene. Sampling along this sphere therefore allows complete capture of the scene with a relatively simple geometry. Circular movement may be implemented by placing the camera on a motorized arm which can be turned around a single axis. The arm allows sampling in one dimension with the second dimension provided by a turntable which rotates the scene relative to the camera. Turntable and arm can be placed in a way that lets the respective axis intersect, effectively providing 2D rotation around a single center point. Splitting the rotational movement between the arm and the turntable also serves to simplify construction, using two completely disjunct parts, which provide the two movements, allowing more a rigid construction with simple means, see Fig. 4.1. A similar approach was also used by Levoy et al. [LH96] and Zobel et al. [ZFS02].

The components available for this work limited the arm length to around 100 cm, giving a maximum diameter for the convex hull of 200 cm, tough smaller diameters are also possible.

With this type of setup the horizontal sampling rate is dictated by the frame rate of the camera and the rotation speed of the turntable, while vertical sampling is obtained by small movements of the arm, which is powered by two rotating spindles, see Section 5.2 for the details of the implementation.

4.4 Optical Setup

As listed in Section 2.2.1, a range of different methods have been proposed for capturing 4D light fields. Specifically there are single exposure methods which capture a 4D light field within a single exposure. The reason why such a camera is not usable for this setup is the large overall amount of light field data to be captured. It is impossible to capture such a large light field with one exposure, due to size limitations both of the optics and the sensor. On the other hand, building a light field from several 4D light field captures does not provide any gains, and reduces image resolution at increased complexity. Specifically the time to capture the full light field, for a given resolution, is bound by the camera bandwidth and independent of the actual method of sampling, under the assumption of similar overhead. Therefore the most simple setup was chosen for this work, which consists of a regular perspective camera, which is moved trough the respective viewpoints while capturing a video stream.

manufacturer	XIMEA
model	MQ042CG-CM
sensor	CMOSIS CMV4000
sensor size	11.27 × 11.27 mm
sensor resolution	2048 px × 2048 px
pixel pitch	5.5 µm
dynamic range (in HDR-mode)	63 dB
frame rate (full resolution)	90 fps
bandwidth	360 MiB s^{-1}
lens mount	c-mount (⌀25.4 mm)
flange focal length	17.526 mm
Bayer filter pattern	RGGB
shutter type	global shutter

Table 4.1: Properties of camera and sensor. Note that dynamic range was measured in HDR mode, see Section 5.4.1.

4.4.1 Camera

The camera used in this is work is a XIMEA MQ042CG-CM and was selected mainly for the high absolute bandwidth, allowing fast recording of the light field, with a maximum bandwidth of 360 MiB s^{-1}. See Table 4.1 for the detailed properties of the camera.

4.4.2 Parameters

The optical system can be described using the pinhole camera model from Section 2.1.2 and the thin lens model from Section 2.1.4. Fig. 4.2 depicts the properties of this setup. To avoid the need for focus stacking, which would incur costly post processing, increase storage requirements and slow down capture, the whole scene has to be captured in focus and with a single exposure. The following sections explain how the parameters of the optical system are derived, while Section 4.4.3 provides the evaluation on this basis.

4.4.2.1 Wavelength

The parameters of the optical setup depend on the used wavelength. Diffraction blur depends on the wavelength according to Eq. (2.9) and the resolution of the lens also depends on the absorption spectrum of the different color channels, see Section 5.1.2. As Bayer sensors introduce additional complexity due to the uneven sampling of different colors, compare Section 2.3.5, the whole setup will be primarily designed with regards to the green color

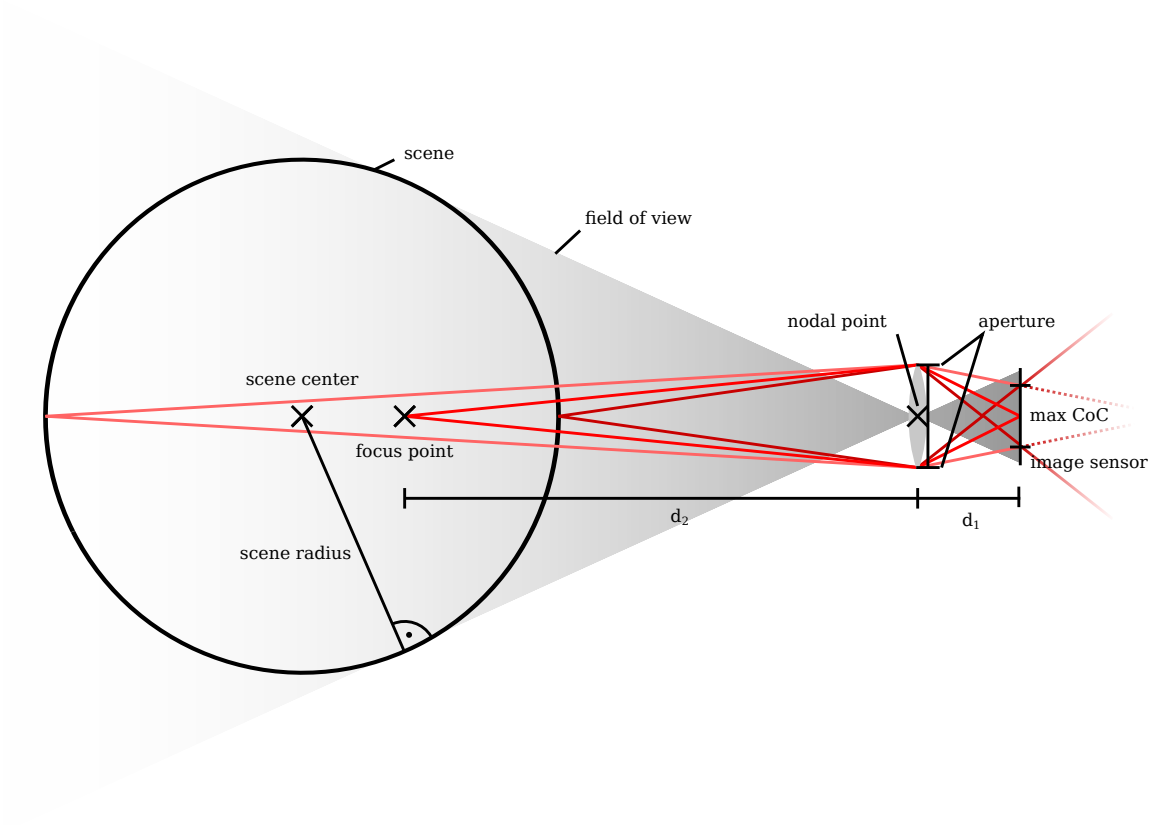


Figure 4.2: Cross section of the optical setup, showing how focus/defocus, indicated by rays in shades of red, is an effect of the aperture and focus point, while field of view, and therefore maximum scene size, depends on the relation between sensor size and focal length. Note that for a given circle of confusion the focus point does not lie in the middle between the two extremes of the depth of field, but moves a bit to the front, see Eq. (2.8)

channel. The reason for selecting the green channel is its higher pixel count and resolution, with double the number of samples than either red or blue. This means the green channel constitutes half the recorded samples, with red and blue representing only a quarter each. Representative for the green channel a wavelength of 550 nm is used in all following calculations, derived from the absorption spectrum of the used camera.

4.4.2.2 Resolution

For the estimation of optical parameters a blur unit of 11 µm is assumed, which equals two times the pixel pitch for the used camera. The reason for this choice is the Bayer pattern of the camera sensor, which doubles the pixel pitch of the red and blue channel, compared to

the raw pixel pitch of the sensor. Basically this is an application of the Rayleigh criterion for resolution, see Section 2.3.4 which is pessimistic in the sense, that the resultant resolution will be very when measured using the MTF50. On one hand this means that the resultant parameters may induce aliasing on the Bayer sensor, on the other hand real lenses also induces additional blur due to imperfections from fabrication and design, making it necessary to confirm any theoretical estimations with measurements of the performance of the actual lens camera combination, as performed in Section 5.1.2.

4.4.2.3 Aperture Size

Diffraction imposes a limit as to how small the aperture can become before diffraction blur sets in. Eq. (2.9) gives the diffraction blur as the radius of the Airy pattern. Using a blur unit of 11 µm gives a radius $r = 5.5\,\mu\text{m}$. The used wavelength is 550 nm, while f denotes the focal length and a the resultant aperture size:

$$a = \frac{1.22 \cdot 550\,\text{nm} \cdot f}{5.5\,\mu\text{m}} = 0.122f \tag{4.1}$$

4.4.2.4 Depth of Field

For the depth of field calculations it is necessary to determine the optimal focus point, as the scene is centered at a fixed position, but depth of field is not the same in front and behind the focus point, see Eq. (2.8) and Fig. 4.2. The optimal focus point is therefore determined by equating the depth of field in front and behind the scene center, with the focus point as parameter, using a numeric method. When regarding the resultant depth of field in Fig. 4.3 the optimal focus point also explains why the depth of field is limited by the scene size. While photographers expect depth of field to approach infinity for small apertures, in this case, as the aperture is closed the scene size increases and the focus point moves closer to the camera, with the focus distance approaching zero, thus balancing front and back depth of field to the same size around the scene center.

4.4.2.5 Field of View

With the optical setup as shown in Fig. 4.2, the scene size s can be calculated using the thin lens formula in Eq. (2.6) and some trigonometry as:

$$s = \frac{w \cdot l}{\sqrt{(\frac{1}{\frac{1}{f}-\frac{1}{d_1}})^2 + (\frac{w}{2})^2}} \tag{4.2}$$

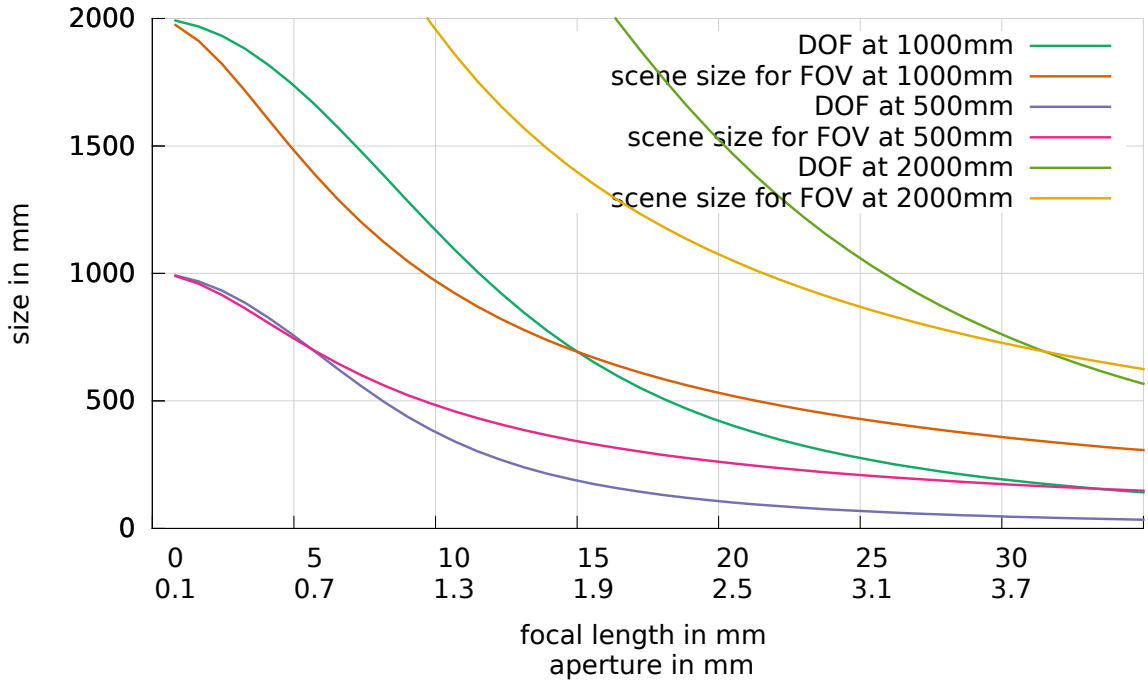


Figure 4.3: Plot of focal length against maximum scene diameter, calculated from depth of field, short DOF, and field of view, short FOV. Both are limited by the setup size. The plot shows three different sizes for the setup, defined by the distance between camera and scene center, see Fig. 4.2. The horizontal axis is given in both focal length and the equivalent aperture according to diffraction, see Section 4.4.2 for parameters and calculations.

The sensor width is given as w, the distance between nodal point and scene center is l, and d_1 is the optimal focus position, derived numerically in Section 4.4.2.4. Note that this means the field of view indirectly depends on the depth of field by connection of the focus point, a relation that becomes apparent when examining the relation between field of view and d_2 in Fig. 4.2.

4.4.3 Evaluation

Fig. 4.3 shows the scene size, limited by the field of view of the camera and by the depth of field, under the constraints stated in Section 4.4.2 and for three sizes of the setup: 500 mm, 1000 mm and 2000 mm. The vertical axis shows the maximum scene size according to the depth of field respectively field of view. The horizontal axis denotes the focal length on top, with the respective aperture size, according to Eq. (4.1) at the bottom.

It is apparent that for each size of the setup there is an intersection point, after which longer focal lengths decrease depth of field faster than the field of view decreases the maximum scene diameter. If full use of the field of view is desired it is therefore necessary to choose a focal length smaller than that at the intersection point, for the respective scene size.

Comparing the different scene sizes, a larger setup gives more leeway as to the usable focal lengths, as the intersection moves to the right. The components available for this work limited the setup to a size of 1000 mm. Therefore a focal length of 12.5 mm was selected to give a bit of headroom. This results in an aperture size, according to Eq. (4.1) of 1.525 mm, which was rounded to $f/8 = 1.5625\,\text{mm}$, as this was the nearest marked aperture size on the lens. This results in a maximum scene size of 811.26 mm and a depth of field of 889.51 mm, see Table 4.2 for the full list of parameters.

camera scene distance	1000 mm
optimal focus point distance	802.19 mm
back focal length	12.30 mm
scene diameter within FOV	811.26 mm
depth of field	889.51 mm

Table 4.2: scene parameters

4.5 Processing and Storage

For processing, a standard off the shelf personal computer was selected, additionally equipped with a dedicated USB3 to PCI-Express adapter, to exploit the full bandwidth of the camera, and with a RAID6 array to provide the required storage capacity and bandwidth. For details on the used components see Section 5.4.

While light field processing requires considerable resources, those are mostly related to the handling of large amounts of raw data. Specifically for this setup the capturing of the light field requires a constant bandwidth of nearly 360 MiB s^{-1} or around 180 MiB s^{-1} after compression.

Hard drives do not provide a constant bandwidth, but rotate with a constant turn rate. The inner tracks of a hard drive therefore provide less bandwidth as they have a smaller diameter and capacity is constant with respect to the track length. Therefore, as hard drives fill up, the bandwidth is reduced. For constant bandwidth applications this means that only the minimal bandwidth is relevant, which lies between 80 MiB s^{-1} and 100 MiB s^{-1} for common hard drives. An alternative is the use of solid states disks which provide significantly higher performance. The problem with solid state disks is the common drop in performance after sustained writes and the significantly higher price for the same capacity.

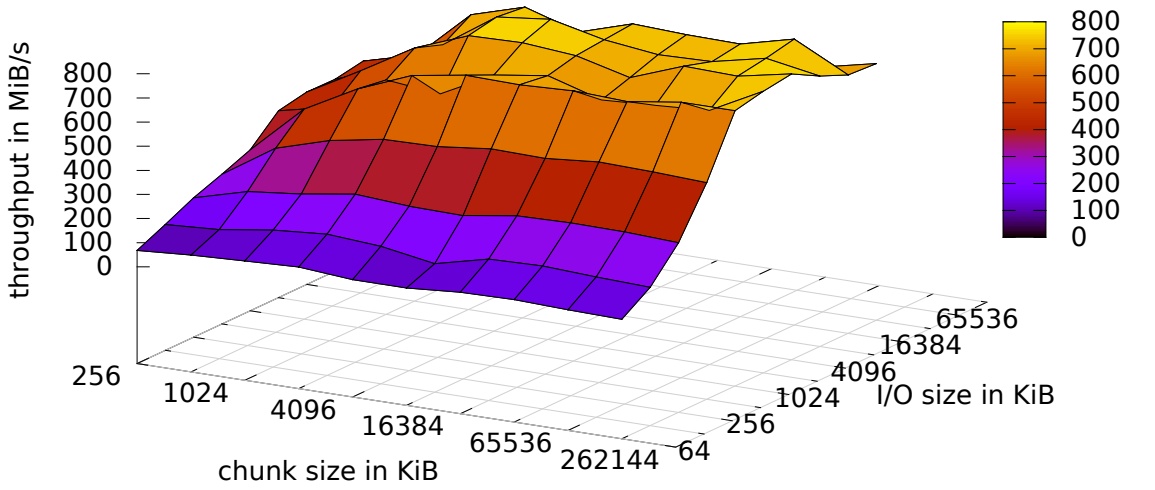


Figure 4.4: Plot for bandwidth of parallel random reads depending on I/O size and raid chunk size for a RAID6 array. A plateau is reached for I/O size $\geq$ 1024 KiB and chunk size $\geq$ 2048 KiB. Measurements were executed at the outer tracks of the array, and therefore represent the maximum values for the tested setup.

As the targeted light field dataset does not fit on neither a single SSD, nor a HDD, a RAID setup is required to achieve the required capacity, which also serves to increase the datarate of the setup, making hard drives feasible even with the low minimal bandwidth. Specifically an array of six 3 TiB hard drives was evaluated for the setup. As the minimal write rate of 367 MiB s^{-1} was more than enough to handle the compressed capture stream, the performance was then optimized for maximum read bandwidth to optimize light field processing.

Fig. 4.4 shows a benchmark of the read bandwidth from the RAID6 array for parallel random reads, for a range of I/O sizes and RAID chunk sizes. The chunk size for the software raid implementation in Linux gives the size into which continuous I/O is split for distribution to the individual devices of the array. From the plot it is apparent that the default value of 512 KiB is a poor choice for parallel reads, independent of the I/O size. Small I/O sizes also decreases performance as the HDDs spend more time reaching the required surface position than for actual reading. Note that for the benchmark the I/O elevator was switched from the default `cfq` to `deadline` with an expiration time of 2 s, which was found to provide better overall performance, and the maximum I/O size was increased to 16 384 KiB from the default of 512 KiB to allow the passing on of the large I/O requests to the underlying block device.

4.6 Compression

Most video and image compression methods, as introduced in Section 3.2, do not reach the performance necessary to process light field data at the required speeds of at least 360 MiB s^{-1} for capture. None reach the performance necessary for processing light field data at the bandwidth of the RAID6 storage array with a read rate of 800 MiB s^{-1} which, as effective bandwidth also has to be multiplied by the compression ratio of at least 2, leads to an effectively required bandwidth, for the decompression, of 1.6 GiB s^{-1}. The fastest method for video compression, `ffvhuff` is just fast enough to enable full speed capture, using at least three processor cores, and without much headroom for other processing tasks. For processing at full I/O bandwidth even `ffvhuff` does not provide enough bandwidth, so other solutions have to be considered.

From the compression methods introduced in Section 3.2 the only class which provides sufficient performance are the 1D bitpacking based methods, for database indices [LB12] and floating point values [BR09]. While those methods are not directly applicable to 8 bit image based light field compression, they demonstrate that bitpacking may be used for very fast compression. The implemented compression method therefore follows the same principles, although significant adaptions were necessary, see Section 5.5 for the specific problems and the implemented solution.

4.7 Calibration

As the mechanical setup in Section 4.3 cannot provide sufficiently accurate position information to describe the individual ray bundles of the recorded light field, some kind of external measurement is required. Given an image to world mapping of individual points it is possible to calculate the extrinsic camera parameters by solving the PnP problem, see Section 2.1.3. Detection of fiducial markers provides location and identification of markers placed in a scene and may be used to provide image to world mappings. State of the art fiducial marker systems provide quite robust detection but are limited in the density at which markers may be placed, and can provide only a limited number of markers, see Section 3.3.

The problems of dense marker placement are summarized in Section 3.3.5. The following lists the properties of a design overcoming those problems, the ordering follows Section 3.3.5.

Error correction: The main difficulty for a dense marker system is the normally incorporated error correction, making markers robust for bit-errors, but increasing marker size considerably, and therefore decreasing the density at which markers may be placed, see Section 3.3. But a dense placement of markers allows the verification of correct marker identification by cross

check of neighboring markers. This requires several markers to be correctly identified and a layout of markers that follows a well defined structure, so neighboring ids may be derived from an identified marker.

Marker count: Replacing the bit error code with a neighborhood based verification reduces marker size, as all coded bits can be used for identification. With a dense pattern in a checkerboard configuration the number of markers can additionally be improved by using the gaps between markers to code a second type of marker as an inversion of the regular type. The payload of the second marker type can then be used as the most significant bits of a combined identification scheme, effectively doubling the number of bits usable for marker identification. For example a marker with an internal bit pattern of 3×3 bit can, with this scheme, code $2^{2 \cdot 3 \cdot 3} = 262144$ distinct markers, compared to only 512 for the regular type.

Orientation fixation: A second use of the error correction code is to serve as orientation fixation. For a marker system without such a code, the direction has to be determined using other means, for example with a gap in the border, present only in one direction.

Corner refinement: Saddle point refinement, like implemented in `CALTag` [AHH10], requires a window for the refinement, which increases the required border size of markers, to avoid interference of other marker structures with the refinement. However if the direction of the lines intersecting at the corner is known, then the window could be shaped so as to include the edges of the marker, but not the central structure, even for a small border.

The marker system implemented according to these principles is detailed in Section 5.6, including a range of smaller improvements. See Section 6.2.2 for results achieved with this system.

5 Implementation

This chapter details the peculiarities of the implementation, realized according to the design from Chapter 4. The problems encountered in realizing those designs are explained, as well as the specific solutions. This chapter also includes more elaborate descriptions for subordinated parts of the designed system, relevant mainly for the specific implementation, but not for the design itself.

The optical setup is regarded first, measuring resolution and deriving the respective sampling rates in Section 5.1, while Section 5.2 describes the mechanical setup, which implements the viewpoint sampling. The contents and the illumination of the recorded light fields are introduced in Section 5.3. The hardware used for capture and processing is detailed in Section 5.4. The following Section 5.5 describes the implementation of the fast compression scheme, developed for fast light field capture but usable for general image compression. The implementation of the high density marker system is detailed in Section 5.6. Lastly the software tools which provide an interface to those implementation, as well as the file format used for storage, is dealt with in Section 5.7.

5.1 Optics and Sampling

The used lens is a 12.5 mm c-mount lens from Edmund optics, selected according to the thin lens model of depth of field and diffraction blur, see Section 4.4. Analysis of the performance of the optical system is crucial for correct sampling, compare Section 2.3.6. But analytical

manufacturer/distributor	Edmund Optics
model	#63-244
lens focal length	12.5 mm
aperture	$f/8$
MTF50 @ $f/8$ (r/g/b)	0.31/0.34/0.32 C/P
field of View	48.5° (65.0° diagonal)

Table 5.1: Characteristics of the lens used in this work. Resolution is given relative to a pixel size of 5.5 µm, and field of view for a square imaging area with a width of 11.27 mm, see Table 4.1.

analysis cannot incorporate all uncertainties of the optical design, manufacture and lens-sensor interactions. The slanted edge method obtains the MTF from recorded test images, and therefore allows analysis of the frequency response of the complete recording system, see Section 2.3.3. Below sections show measured lens resolution with a comparison to an optimal diffraction limited lens, simulated with the same parameters. Afterwards, the sampling rate is derived using the measured resolution and the scene geometry, according to Section 2.3.7.

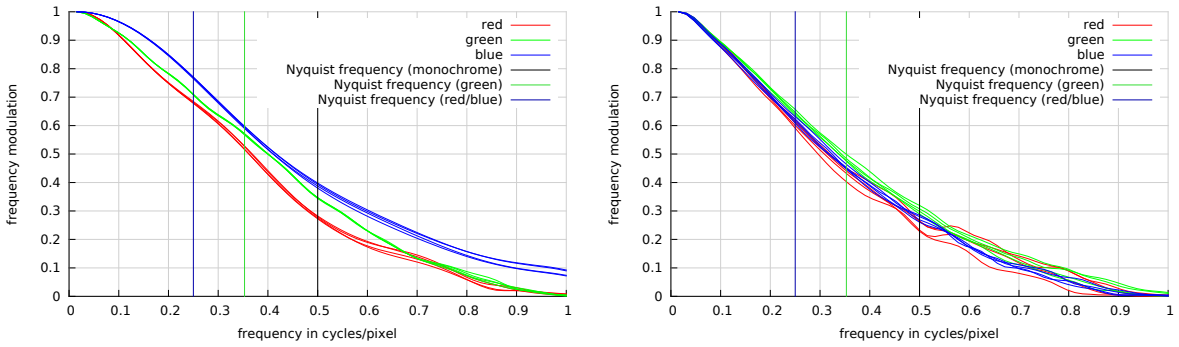
5.1.1 Parameters

A summary of the lens characteristics is available in Table 5.1. For sufficient depth of field the aperture was fixed at $f/8 = 1.5625\,\text{mm}$, see Section 4.4.3, and focus distance set as close as possible to the optimum of 802.19 mm.

5.1.2 Resolution Simulation and Measurements

Lens resolution was measured with the selected parameters using the open source `mtfmapper` software [Ber] and results are shown in Fig. 5.1, together with a simulation of an optimal diffraction limited lens. Both results were obtained using the slanted edge method as implemented in `mtfmapper`. For the simulated result an image was rendered using the `mtf_generate_rectangle` tool, simulating pixel geometry and diffraction at three wavelengths and merging the three renderings into a simulated Bayer pattern. Pixel geometry and micro-lens configuration was not available so square pixels with 100% fill rate were assumed.

In Fig. 5.1a the MTF curves for the simulated lens are sorted by wavelength, as expected for a diffraction limited system, where longer wavelengths induce more blur according to Eq. (2.9). However Fig. 5.1b shows the results for the real lens, where the green channel achieves the highest resolution, followed by blue and red. While this ordering cannot be explained by diffraction there are several explanations, which may jointly produce the observed effect. First, lenses are primarily optimized for green wavelengths, as the human visual system is most sensitive to those, and Bayer sensors, for this reason, have normally as many green photo diodes as red and blue combined. Furthermore, from the camera datasheet it is apparent that the absorption spectrum for the green channel is notably smaller than for red and blue. As the MTF measurements are derived from a black and white chart, and refraction is depended on the wavelength, blur is therefore depended on the spectrum, with a larger absorption spectrum inducing more blur, as the extrema of the spectrum experience more varying refraction within the lens.



(a) Simulated MTF response incorporating diffraction and square pixels.

(b) Measured MTF obtained using the slanted edge method.

Figure 5.1: Two plots showing simulated and measured frequency modulation, with a focal length of $f = 12.5\,\text{mm}$ and an aperture of $f/8$. The vertical axis gives the frequency modulation and the horizontal axis the frequency in cycles per pixel. MTF50 gives resolution as the frequency at which attenuation reaches $-3\,\text{dB}$, respectively a modulation with a factor of 0.5. The simulated results show the expected behavior of a diffraction limited lens, as shorter wavelengths obtain higher resolution, while the actual lens used in this work, obtains a lower resolution, due to imperfection in the lens. The Nyquist frequency for such a plot is always 0.5 C/P when regarding the full resolution, half the frequency of the sampling. However, for Bayer sensors the sampling must also be regarded separately for individual color channels, giving a Nyquist frequency of $\frac{1}{\sqrt{2}} \cdot 0.5\,\text{C/P} = 0.35\,\text{C/P}$ for the green channels due to the diagonal layout, and of $0.5 \cdot 0.5\,\text{C/P} = 0.25\,\text{C/P}$ for the red respectively blue channel.

An aperture of $f/16$ was also evaluated, resulting in a plot more in line with a diffraction limited lens, with MTF curves ordered by wavelength. Overall obtained resolution was accordingly lower than at $f/8$. The use of $f/16$ was not practical for the final recording as light loss due to the small aperture was too high, requiring longer exposure times and therefore a lower capture rate, increasing overall capture times. The results at $f/16$ are available in Appendix A.2.

5.1.3 Spatial Sampling

Fig. 5.1b gives average MTF50 values of 0.31/0.34/0.32 C/P for the three color channels red/green/blue. With a scene diameter of 70 cm and the distance from the scene center to the camera at 100 cm, it is now possible to calculate the required sampling rate. The maximum resolution of the camera lens combination according to the MTF50, see Section 2.3.4, is

reached in the green channel with:

$$\text{MTF50}_{green} = 0.34\,\text{C/P} = \frac{0.34}{5.5\,\mu\text{m}} = 61.81\,\text{Lp/mm}$$

The unit Lp/mm represents line pairs per millimeter, a measure used in optics. A line pair is simply one period. This means Lp/mm could simply be written as mm^{-1}, Lp/mm is used to denote that the frequency represents a spatial resolution.

The maximum magnification is given by the minimal camera scene distance of $1000\,\text{mm} - 0.5 \cdot 700\,\text{mm} = 650\,\text{mm}$ and the focal length of $f = 12.5\,\text{mm}$, according to Eq. (2.7):

$$m = \frac{12.5}{650} = 0.0192$$

To derive the sampling rate, the MTF50 is used as the cutoff frequency on the image plane. By reverting the magnification m experienced in the transformation from scene/world to image space, the cutoff frequency is backprojected from the image space into the scene, giving the actual spatial cutoff frequency f_{cw} in the scene:

$$f_{cw} = \text{MTF50}_{green} \cdot m = 1.189\,\text{Lp/mm}$$

The maximum viewpoint sampling distance $d_{spatial}$ which provides sufficient spatial sampling is the period of $2 \cdot f_{cw}$, according to the sampling theorem in Section 2.3:

$$d_{spatial} = \frac{1}{2f_{cw}} = \frac{1}{2 \cdot 1.189\,\text{Lp/mm}} = 420\,\mu\text{m}$$

5.1.4 Angular Sampling

To calculate the maximum viewpoint distance for correct sampling of the angular components, a small angle approximation is used, as the focus distance of $d_1 = 802.19\,\text{mm}$ is much larger than the aperture $a = 1.5625$, see Section 2.3.8. Using Eq. (2.12) this results in a maximum sampling distance d_{ang} of:

$$d_{ang} = 0.83a \approx 1.3\,\text{mm}$$

This is quite a bit larger than the spacing required by the spatial resolution, and hence not relevant for this configuration of the system.

5.1.5 Bayer Pattern Sampling

The camera records color images using a Bayer sensor, recording only one color channel per pixel, see Section 2.3.5. While the viewpoint distance induced by the spatial and angular resolution can be accommodated by using smaller movements of the mechanical setup, the

Bayer sensor may still obtain aliasing. The required on-sensor sample distance may be derived, per channel, from the measured resolutions as $d_{r/g/b}$:

$$d_r = \frac{5.5\,\mu\text{m}}{2 \cdot 0.31} = 8.87\,\mu\text{m}$$
$$d_g = \frac{5.5\,\mu\text{m}}{2 \cdot 0.34} = 8.09\,\mu\text{m}$$
$$d_b = \frac{5.5\,\mu\text{m}}{2 \cdot 0.32} = 8.59\,\mu\text{m}$$

While the sensor has a pixel pitch of $5.5\,\mu\text{m}$, the red and blue samples are only repeated every two pixel in both dimensions, leading to a sampling period of $11\,\mu\text{m}$. Green is sampled every second pixel, see Fig. 2.6, for an effective sampling period, using the diagonal, of $\sqrt{2} \cdot 11\,\mu\text{m} = 7.78\,\mu\text{m}$. This means that the red and blue channel may exhibit aliasing, while the green channel does not. This problem could be solved by closing the aperture further, inducing more diffraction blur, or by using an optical lowpass. However, further closure of the aperture would reduce the amount of light reaching the sensor, inducing more noise for the same exposure. On the other hand an optical lowpass filter cannot simply be attached to the lens, but is placed directly in front of the sensor, making installation in an existing camera difficult. For this reason none of those solutions was implemented. On the other hand, the used configuration allows the comparative evaluation of aliasing effects, by comparing the green channel which should not exhibit significant aliasing, to the red and blue channel. Indeed the effects of aliasing are clearly visible in some renderings from the dataset, see Section 6.3.4.

5.2 Mechanical Setup

The mechanical setup is based around a computerized turntable and arm, see Section 4.3 for the concept. This setup was implemented in two parts, using `OpenBeam` [OPE03] extruded aluminum profiles for structural support supplemented by fischertechnik [FT03] parts for movement and assembly. The turntable was based on a single large gear ring placed under a wooden plate on which the scene is placed. Both parts of the setup are independent, the turntable is placed in the center of the structure supporting the arm, so that the respective rotation axis intersect, in effect providing rotation around a single point in two directions. Fig. 5.2 shows the implemented setup, highlighting the support structure and the arm, while Fig. 5.3 shows the construction of the turntable.

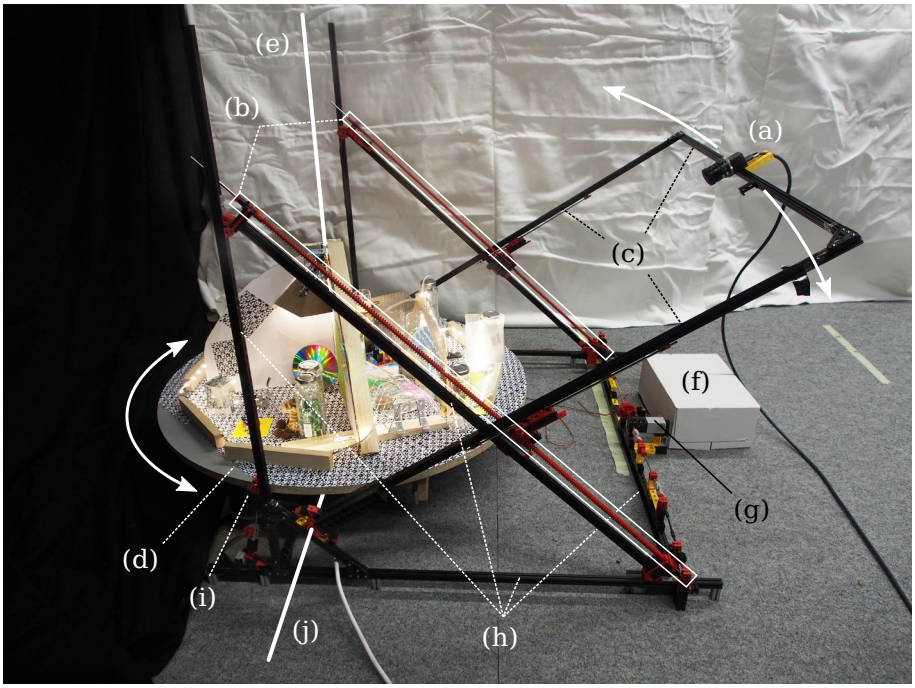


Figure 5.2: An image of the complete setup. The scene on the turntable (d) rotates around a vertical axis (e), and is capture by the camera (a). The camera is attached to an arm (c), which moves it around a horizontal axis (j) via two spindles (b), powered by a motor (g). The setup is controlled via a single board computer in (f). A friction wheel smooths out the turntable rotation (e).

5.2.1 Power and Control

The setup is powered by two electric motors, which are driven by a lab power supply and controlled with a STMicroelectronics L298N dual H-bridge motor driver, itself controlled via the GPIO ports of a Raspberry Pi [RBP03] single board computer, see Section 5.7.5 below for the software details of motor control.

5.2.2 Arm

Fig. 5.2 shows the assembled setup with all important parts marked. The “arm” of the gantry (c) is constructed from three 100 cm aluminum profiles connected to form a “U”. The two ends of the arm are connected over two short axis with the frame of the setup, allowing up and down rotation around a horizontal axis (j), but leaving a large space in the middle of the frame for the turntable (d). Movement of the arm is executed via two long spindles (b), powered by a single motor placed in between (g). To provide the small incremental movements of the arm

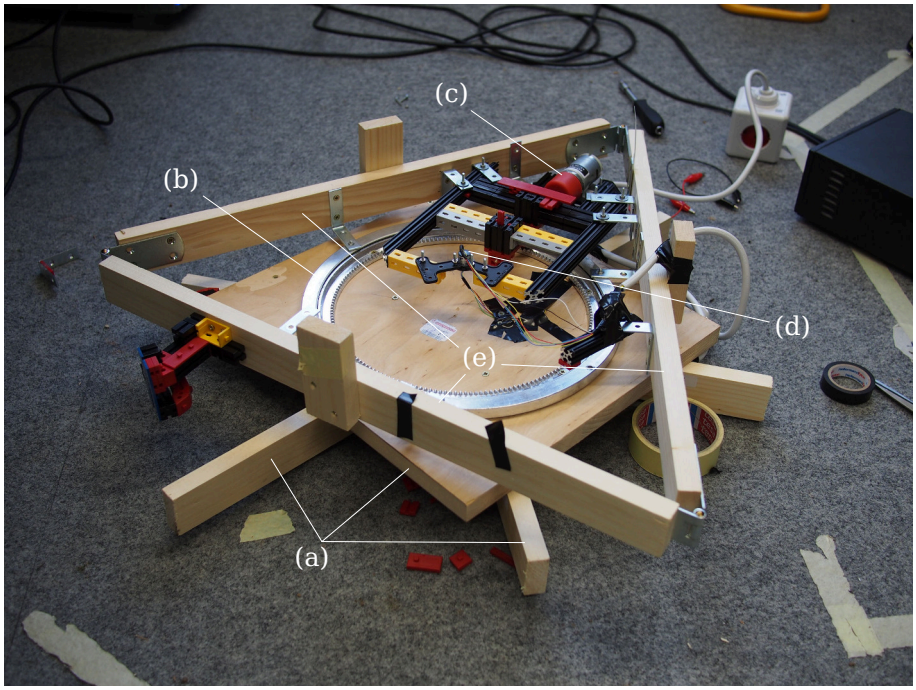


Figure 5.3: The turntable used for rotating the scene, here displayed upside down to show the components. The motor (c) drives the large ring gear (b) to rotate the carrier and base plate (a), against the support structure which stands on the floor (e). A slip ring provides power to the turntable (d).

for dense light field sampling, the motor is only powered for a short pulse, the time of which is determined using trial and error, to arrive at the required movement for dense sampling as calculated in Section 5.1 above.

5.2.3 Turntable

The turntable allows rotation of the scene to sample a single horizontal slice of the light field, see (d) in Fig. 5.2 for the assembled turntable and scene. In Fig. 5.3 the turntable is shown bottom up with the main parts marked, and without the large plate which would be placed on the rotating base plate and carriers. Carriers and base plate (a) are connected to the non-rotating support structure (e) via a large gear ring (b) which is turned by an electric motor (c) with built in gearbox with a high reduction of 3000:1, to provide the required slow rotation speed. As illumination has to be provided from within the scene, a slip ring (d) is installed in the center to relay the required power to the rotating part of the turntable. Like for the gantry, speed of the turntable is adjusted using trial and error to provide the required speed for the sampling rate calculated in Section 5.1. Turntable movement can sometimes be a

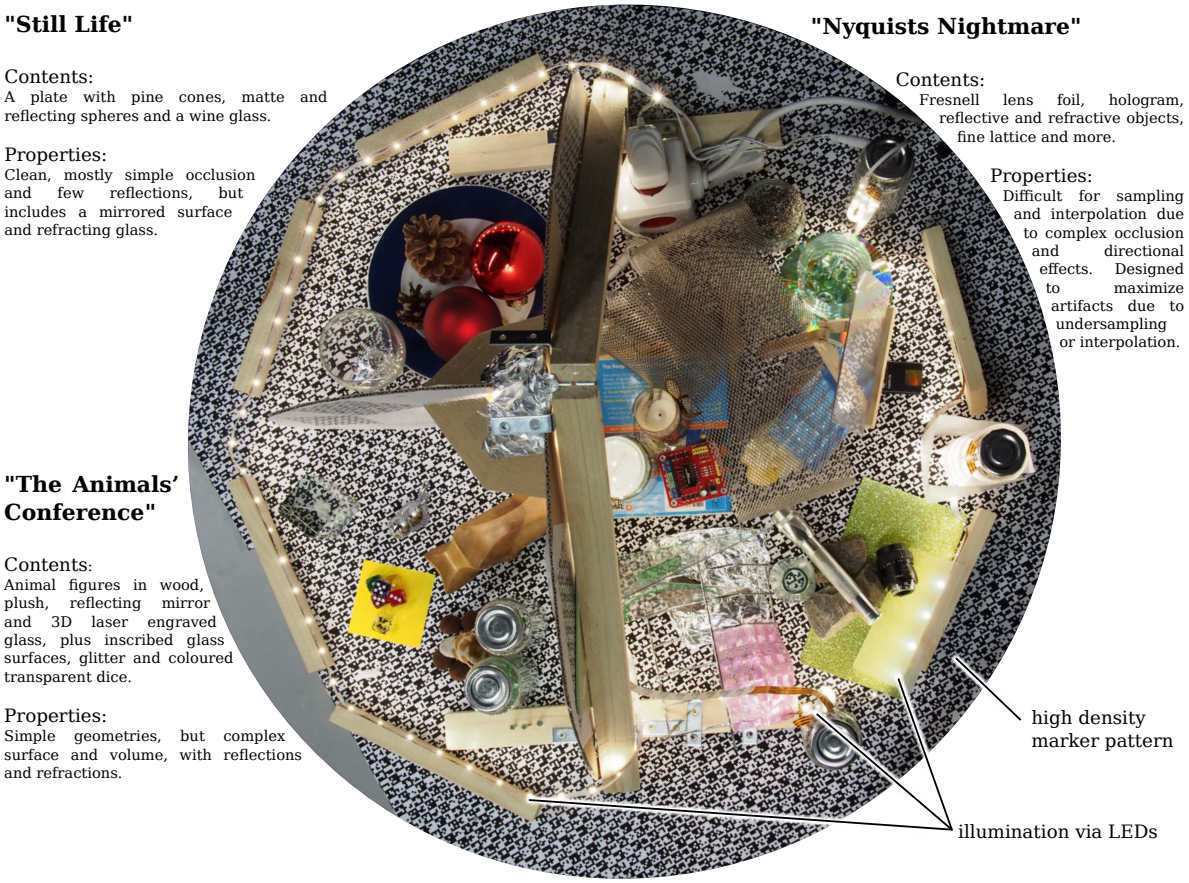


Figure 5.4: Overview of the assembled scene. Illumination is provided trough 6 m of LED strips. The scene is split into three compartments, a brief description of contents and properties is provided besides the respective compartment. For an extensive list see Appendix A.3.

bit irregular, when small resistances in the gear ring are overcome by the motor. Therefore continuous friction resistance via a small rubber wheel was added to the turntable, see (i) in Fig. 5.2, providing a consistent resistance and therefore smoothing the turntable movement.

5.3 Scene

With a diameter of 90 cm the turntable is not completely occupied by the scene, as it does not completely fit into the field of view of the camera, see Section 4.4.3. The actual scene has a diameter of around 70 cm, however illumination, which may be considered part of the scene, slightly protrudes from this sphere. Fig. 5.4 gives an overview of the turntable, showing the scene and illumination.

5.3.1 Illumination

The illumination of the the scene is powered trough the slip ring integrated in the turntable, see Fig. 5.3, and provided by 6 m of LED strips, giving even illumination to all parts of the scene, but still providing individual highlights and directional effects for a more challenging light field. Fig. 5.4 shows the illumination as small lights surrounding the scene. The reason to use LEDs, apart from the low power consumption, is the low risk of failure or deterioration of illumination over the course of a long recording. An earlier version of the assembly included lights with hue control, which could have allowed multi-spectral recording using several recording passes, or full camera resolution without color sub sampling of the Bayer pattern, using single color illumination. However those lights were incompatible with the short partial exposure times of the HDR mode of the sensor, see Section 5.4.1, due to used pulse width modulation used for color control, which resulted in unpredictable brightness changes.

5.3.2 Contents

The optical setup in Section 4.4 results in a relatively large scene, allowing a high number of objects and light interactions to be placed in a single recording. This also gives the possibility to split the full scene into several smaller scenes, placed in different compartments. The compartments are separated by a wall to avoid interaction between the sub-scenes. The turntable was split into three compartments, see Fig. 5.4. The biggest compartment spans half of the turntable providing the full depth of the scene for occlusions. The first small compartment contains a clean scene composed of a plate with a few pine cones, a matte and a reflecting sphere as well as a wine glass. The second compartment contains more difficult objects, including reflecting and refracting objects, small font on transparent surfaces and a glittering surface, exhibiting very directional characteristics. The largest compartment contains all difficult and optically complex objects that were available, including a lattice for complex occlusion, Fresnel lens foil, a lenticular lens and screen, surfaces with complex BRDFs like a hologram, glitter, a mirror, a Compact Disk and more. For an exhaustive description see Appendix A.3. The diameter of the scene is roughly 70 cm, staying within the limits defined by the geometry of the setup, however the turntable is slightly larger and therefore not completely visible in the recording.

5.4 Capture and Processing Hardware

The capture and and processing hardware is mostly made up of generic off-the-shelf components, aside from the camera which was select for the high absolute bandwidth, allowing fast light field capture, see Section 5.4.1. The storage and processing is introduced in Section 5.4.2.

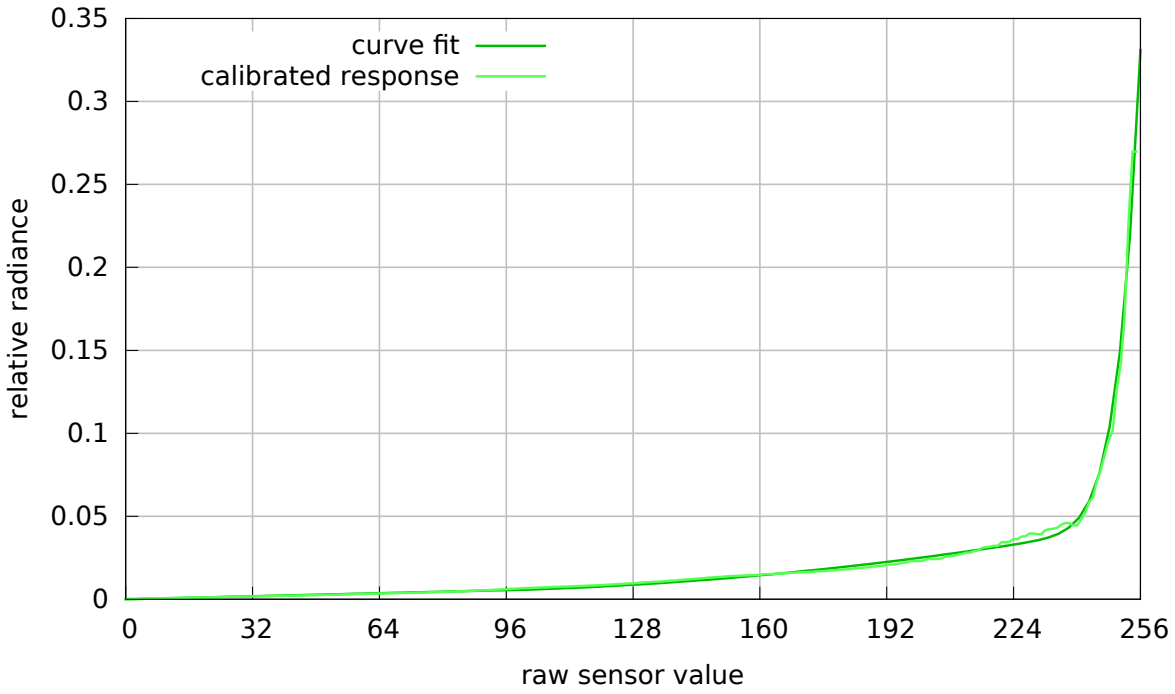


Figure 5.5: Plot of the inverse camera response curve, used to determine a relative radiance from the measured values on a linear scale. Not that radiance here is expressed as the relative radiance, due to the missing absolute reference frame. The curves were calculated using `pfstools` camera calibration.

5.4.1 Camera

The camera used in the setup is a XIMEA MQ042CG-CM, see Table 4.1 for technical details. Because of the small aperture used for this setup, as well as the short exposure time the gain was set to 3 dB for capture. The camera does not include dark frame subtraction, therefore after capture a darkframe was acquired with the same temperature of 42 °C as was reached during capture. Nevertheless due to the high temperature and gain, noise is quite prominent for the chosen settings, even after darkframe subtraction, see Fig. 6.9. Because multi exposure high dynamic range capture was not possible due to space constraints, the HDR mode of the sensor was evaluated and used in the final capture. This mode operates by providing an interface, which allows to schedule several sub-exposures for each full exposure, while also limiting the saturation level of the sensor to a specified level for each sub-exposure. This allows the implementation of non-linear sensor response, by limiting saturation to some fraction of the full well capacity for a long exposure time and then successively raising the saturation limit while exposing for shorter durations. The sensor allows for a total of three such exposures, which are combined on the sensor to provide the final non-linear image. As the camera does not allow full bandwidth capture at a higher bit depth than 8 bit, HDR capture was used with

the low bit depth of 8 bit. Non-linear sensor response still provides an effective dynamic range of 63 dB, even in the presence of increased noise for the HDR mode. The camera response was calculated using `pfstools` [MKMS07]. Fig. 5.5 shows the inverse camera response curve, displaying the non-linearity.

5.4.2 Transmission and Storage

The personal computer used for capturing and processing of the light field dataset is based on an Intel® Core™ i7-860 Processor running at 2.8 GHz, installed on an ASUS P7P55D PRO motherboard and accompanied by 8 GiB of DDR3 memory. Storage is supplied using 6× 3 TiB Seagate Barracuda 7200.14 ST3000DM001 hard drives in a RAID6 configuration providing a minimal sequential write speed of 367 MiB s^{-1} and a read speed of up to 800 MiB s^{-1}. Implementations of he USB3 standard differ in the maximal achievable bandwidth and the on-board USB3 adapter only provides around 160 MiB s^{-1}. Therefore a dedicated Inateck KTU3FR-2O2I USB3 to PCI-Express adapter was added, which reaches a speed of 280 MiB s^{-1} when placed in one of the secondary PCI-E ports. When placing the adapter in the PCI-E port designated for the GPU and the GPU in a secondary port the system reaches a speed of 320 MiB s^{-1} and with the GPU completely absent from the system 344 MiB s^{-1} are possible, nearly the nominal maximum bandwidth of the camera. However, the camera respectively the USB3 driver was still susceptible to dropped frames in the presence of even small delays caused by OS scheduling and even CPU frequency transitions. Therefore all CPU frequency scaling (Turbo Boost and SpeedStep) was disabled and processing was kept to a minimum, with compression and recording not fully utilizing even a single CPU core. Also the recording software was run with realtime priority, including the processing tasks started by the camera driver, resulting in around 0.2% dropped frames.

5.5 Compression

From the fast compression methods introduced in Section 3.2, the SIMD based integer compression method by Lemire and Boytsov [LB12] is the fastest by a wide margin, which is why it was selected for further evaluation. At the same time it is not directly suited for image compression, due to the implementation using 32 bit integers, and a direct port to 8 bit image coding suffers from the following problems:

Less latitude: The recorded light field data has a bit depth of 8 bit compared to at least 32bit in database indices, so each bit utilized has four times the impact on compression performance.

Large block size: While SIMD on x86 has a width of 16 byte, for performance reasons the implementation in [LB12] uses a block size of 128 integers, respectively 512 bytes. Image data has a much higher variance compared to sorted database indices, especially considering the smaller range for 8 bit data, therefore a smaller block size is required for efficient compression. A smaller block size however decreases performance and increases signaling overhead, which in turn is problematic due of the higher impact of spent bits on compression performance, see above. A smaller block size also increases the impact of constant per block calculations, further reducing speed.

Missing SIMD instructions: On x86, many SIMD instruction that are available for the processing of 32 bit integers, as for example shifts, are not available in byte variants and have to be replaced by more expensive combinations of 32 bit shift and mask operations.

Size increase due to delta coding: Pixels in an image are not sorted by size, as is the case for database indices. Therefore deltas between consecutive pixels in images require one extra sign bit, an overhead of 12.5 %.

5.5.1 Overview

The approach revolves around bitpacking, which is the compression of integers by storing only the significant bits and the number of bits required. If the number of significant bits is calculated and stored individually for each number, then this represents a universal code, see Section 2.4.1.1. To facilitate vectorization and avoid excessive space use due to the coding of the significant bit counts, the bitpacking is applied to whole blocks of bytes, with larger block sizes resulting in higher speed at reduced compression efficiency. This approach and this implementation will be referenced simply by the name of block wise bitpacking, short `BBP`.

Typical image data is spatially correlated. Lossless image compression methods exploit this correlation by predicting pixel values from previously coded neighbors and coding of the residual, the difference between prediction and actually observed value. An example for such a predictor is the median edge predictor in `jpeg-ls`, which uses three neighboring pixels for prediction. In contrast to those schemes `BBP` uses only one dimension for prediction and

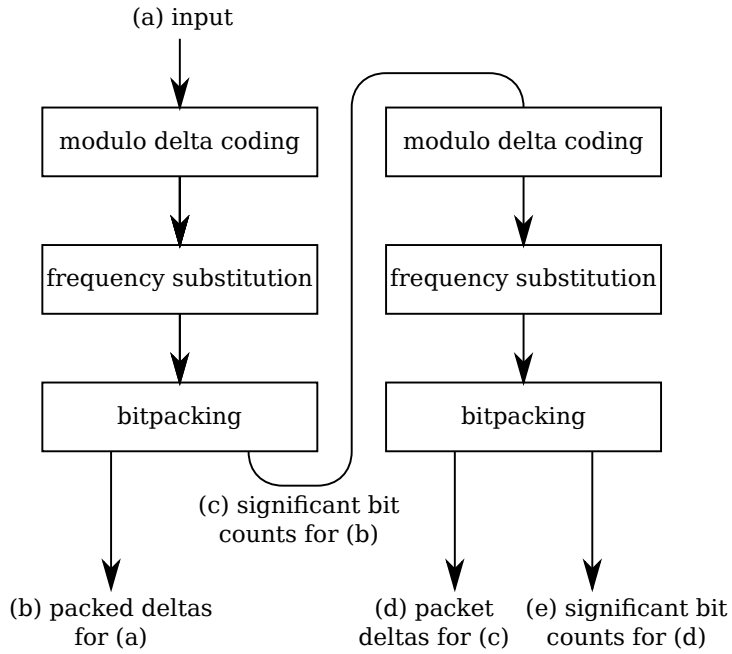


Figure 5.6: Flow chart illustrating the compression procedure. From the input (a) the process produces two streams, the packed data (b) and the data signaling significant bit counts (c). As significant bit counts are itself still compressible, the process is repeated resulting in streams (d) and (e). The output of the whole procedure are the streams (b), (d) and (e). For decoding (c) is reconstructed from (d) and (e), to provide the significant bit counts to decode (b).

the other to allow SIMD implementation of the bitpacking routines. This represents a trade off, of coding efficiency for a higher speed. Compared to Huffman or arithmetic coding, see Section 2.4.1, bitpacking cannot adapt to the distribution of symbols. Frequency substitution is used to place higher frequency values in codes with a low number of significant bits, see Section 5.5.4. Also bitpacking results in a variable length code which is not prefix free, therefore it is necessary to know the number of significant bits before a block can be decoded. Those values have to be signaled separately. The need to store the number of significant bits is another reason, apart from vectorization, for the division into blocks. If the block size approaches one byte the overhead of signaling the number of significant bits outweighs the increase in coding efficiency. To reduce efficiency loss for small block sizes, and therefore allow higher overall efficiency, the whole scheme is recursively applied to the significant bit counts, produced by a first compression pass, see Fig. 5.6 for an overview of the whole scheme.

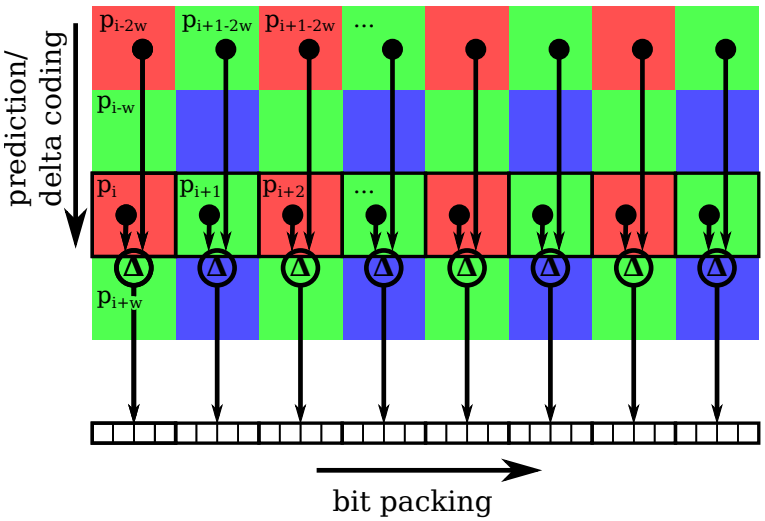


Figure 5.7: Illustration of vertical prediction and delta coding and the horizontal bitpacking. For a Bayer pattern image the prediction for pixel p_i is executed from samples two lines above: p_{i-2w}, where w is the image width. Predicted pixels are shown with a black border. The calculated bit pattern is then packed in horizontal blocks, illustrated at the bottom as white boxes.

5.5.2 1D Predictor

Delta coding can be regarded as the simplest form of prediction, each sample is simple predicted to have the same value as the last one, so calculation of the residual simplifies to the calculation of the difference between two samples. Compared to database indices, image data is correlated in two dimension. Normally this is exploited to improve compression efficiency using a 2D predictor, as for example in `jpeg-ls`. But in `BBP` this correlation is used to accelerate performance by coding the delta not between horizontal neighbors but vertically, followed by horizontal bitpacking. Both these steps can be implemented with SIMD semantics, resulting in a high speed. The correlation in the vertical direction gives residuals that are smaller than the actual sample values, while horizontal correlation means that a single block tends to group samples with similar significant bit lengths, reducing efficiency loss due to block-wise handling.

Normally, decoding a sequence of deltas is problematic for SIMD, as it requires the calculation of the prefix sum, which requires additional steps for a parallel implementation. In [LB12] an offset of 16 Bytes is used, the width of SIMD, which avoids the reference to decoded elements within a single SIMD instructions, but induces a loss off efficiency. Lemire and Boytsov [LB12] report four times larger deltas by moving the prediction from a distance of one integer to

four. The vertical prediction scheme used in BBP on the other hand allows the use of direct neighbors, as a single instruction can operate between two lines of the images, avoiding the need for parallel prefix sum calculation trough the 2D layout of the image.

This method also gives more flexibility for the layout of the input data. Because of the in-memory layout of images as continuous chunks of memory, vertical prediction is implemented using a fixed offset. If the input data consists of interleaved samples, for example rgb images or raw Bayer patterns as produced by the dense light field capture system, then the offset may be adjusted so that prediction is always executed from the same sample type. For example for Bayer data, the offset is set to two times the image width, which works for any pattern that repeats every two lines. This approach further increases performance because no preprocessing steps are necessary for a wide range of inputs. Fig. 5.7 illustrates the interaction between vertical prediction and horizontal bitpacking, on the example of a Bayer pattern image.

5.5.3 Modulo Delta Coding

For 8 bit values the difference between prediction and observed value may be anywhere between -255 and 255, a range which does not fit into 8 bits. To avoid the necessity to expand the coding to 9 bit, which would half the effective SIMD width and waste one bit of space, modular arithmetics are utilized. As $\mathbb{Z}/256\mathbb{Z}$ is a commutative ring, the inverse is calculated just the same as for regular subtraction. Implementing modular arithmetics is trivial, as wrapping at the maximum value is the default mode of operation for non-saturated integer math on x86.

5.5.4 Frequency Substitution

While small differences are very common for delta coding in images, and small values are well suited for bitpacking, the use of non-saturated wrapping arithmetics maps small differences to large values, like -1 to 255, which requires the full 8 bit to store. To accommodate for this, frequency substitution is applied, replacing the frequent, small values with bit codes which use few significant bits. This is equivalent to an ordering of the values by the minimum absolute value of the two possible deltas before the modulo operation: 0/0 stays 0, -1/255 maps to 1, -255/1 maps to 2, -2/254 maps to 3, and so on. Fig. 5.8 illustrates the procedure. This mapping describes a triangle function with a slope of 2, where the first slope maps to even values and the second slope to odd values. This function is simple to implement with SIMD instructions, leading to a much faster per pixel performance as compared to a look up table.

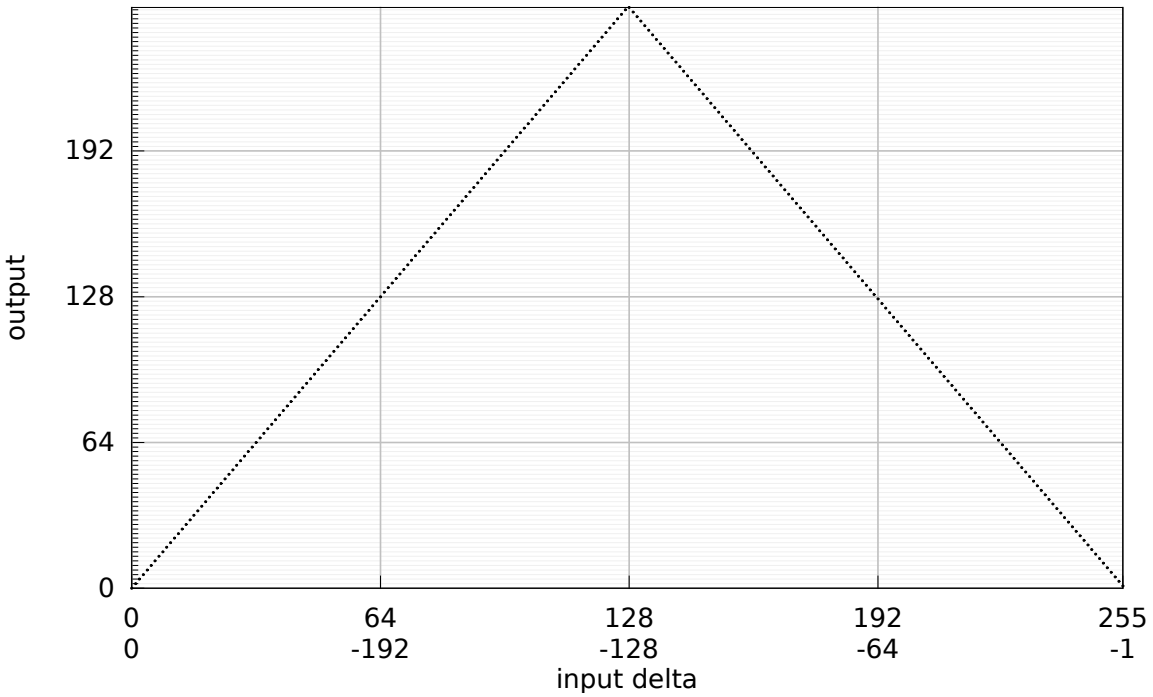


Figure 5.8: The function used for the frequency substitution is a triangle function, the horizontal axis denotes the input values, from -255 to 255, but coded using 8 bits using modular arithmetics. Even and odd values interleave the input according to the sign of the smallest distance from zero. This is visible when looking at the minor grid lines, which are draw for even values only. All data points of the first slope lie on the minor grid lines, while for the second slope they lie in between.

5.5.5 Significant Bit Count

To get the maximum significant bit count for any byte in a block, all bytes of the block are merged using bitwise OR, and significant bits are determined from the result, using a count leading zeros operation, if available, or simply a look up table. Bitwise OR is associative and well suited for SIMD. The implementation makes full use of this fact, processing several blocks at the same time to achieve high speed.

5.5.6 Block Wise Interleaved Bitpacking

The packing uses a vertical layout where a block of n bytes is interleaved into a block of the same size, with unused bits remaining at the same position in every byte. Consecutive blocks are interleaved into the unused bits of each byte until no unused bits remain, which leads to the write out of the current block and allocation of the next one. See Fig. 5.9 for a visualization of

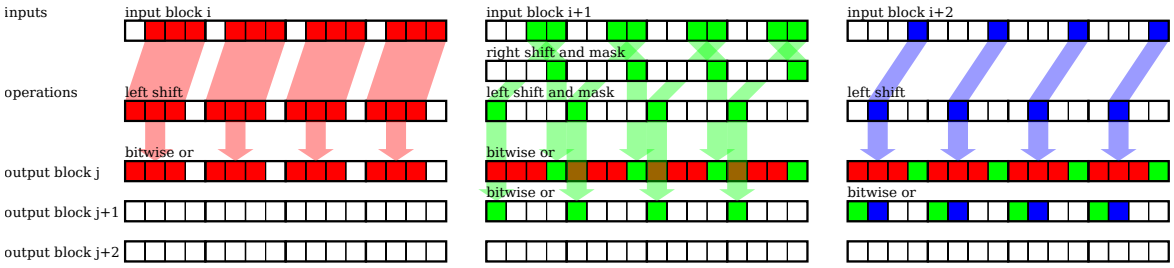


Figure 5.9: Illustration of the interleaved bitpacking scheme. For clarity the depicted blocks have a size of four samples with four bit each, instead of 8 bit as in the actual implementation. Bits are represented by boxes, significant bits are colored, white blocks denote a bit with a value of zero. The packing routine distinguishes two cases, either there is enough space to store the significant bits as is the case with block i and i+2, or the block has to be split between the current and next output block as with block i+1 in the illustration. The colored arrows denote executed operations. Note that the operations are always performed with the whole SIMD width and not per element.

the procedure. Lemire at Boytsov [LB12] use a computed jump to one of the several branchless implementations of bitpacking routines, one for each combination of required and available bits, which is difficult to predict by the CPU. Compared to [LB12], this implementation only branches over the block full condition and processes any bit combination using the same code, with a computed bit mask and shift. This branch is the only one within the compression loop which reduces misprediction rates. This results in a lower penalty for small block sizes which is necessary for efficient image compression. Because the bitpacking relies only on shift and mask instructions it is possible to provide pseudo-SIMD versions with 4 and 8 Bytes, which do not rely on specific CPU instructions but simply utilize regular 32/64 bit arithmetic to process all bytes of a block at the same time. These variants can be used to provide vectorization on CPUs that do not support explicit SIMD.

5.5.7 Recursive Application

For small block sizes and/or high compression ratios the number of bits used for signaling the bit length of the packed blocks becomes non negligible. For example with a block length of 8 bytes and a compression ration of 1:4 at least 4 bits are necessary to code the significant bit counts. For the compressed block size of 16 bit this amounts to an overhead of over 25%. To improve compression, the full compression scheme is applied again, but only to the data signaling significant bit counts. This approach also simplifies the implementation of the significant bit handling, because signaling data does not have to be packed into a bitstream but is processed simply as bytes. The recursive approach is illustrated in Fig. 5.6.

5.5.8 Inter Coding

An additional mode for the compression of video and 3D light field data was also evaluated. The mode is an inter frame compression mode, exploiting inter frame correlation between consecutive frames. In this mode the `BBP` compression scheme, as described above, is applied to individual frames in the same way as before. But as a preprocessing step, consecutive frames are processed with the modulo delta coder from Section 5.5.3, by calculating the delta between the same pixels in consecutive frames, but without frequency substitution. Those deltas are then passed as frames to the regular `BBP` coder. This makes use of the additional correlation within the video and light field data. Compression ratio is slightly improved but at the cost of slower compression speeds, see Section 6.1 for results.

5.6 High Density Fiducial Markers for Metric Calibration

The capture setup as described in Section 5.2 does not provide more than a very rough estimate for the position of the camera. The turntable position can only be determined at one point for each turn. Implementing the same setup with high quality parts with high mechanical precision and small enough slack to actually keep that precision over the full range of motions, is a challenging task. Fortunately, if incremental movement is precise enough that a minimal sampling interval can be guaranteed, then the system is able to capture a dense light field, and extrinsic calibration of the camera can be performed in an extra step.

To avoid the necessity to measure the scene by external means, a printed marker pattern is used, which can easily be applied to a sufficiently flat surface to produce a precise target of known dimensions. Fiducial marker system are normally optimized to allow reliable recognition of individual markers at a high speed, which is useful for augmented reality applications. see Section 3.3. This includes increasing the marker size to include high quality error correction codes. However for camera calibration a high number of very precise markers is more useful. Therefore the developed marker system is optimized to provide many precise calibration points by reducing the marker size and removing the need for explicit error correcting codes, using a layout where any detected marker is correlated with its neighbors to detect errors.

To simplify production and recognition, the markers introduced here use only black and white elements. Colors might be useful to increase the number of distinct markers, but at a cost with regard to robustness in the presence of difficult illumination. See Section 6.2.1 for a demonstration of the marker detection.

5.6.1 Marker Structure

The marker system introduced in the following is named `HDMarker`, for the target of a high density of markers. The following sections first introduce the structure of a single marker and then the dense placement of multiple markers.

5.6.1.1 Marker Structure

The crucial property, determining the density at which markers may be placed, is the size of the smallest element within the marker. If the smallest elements in the marker becomes so small that they cannot be distinguished, then even error correction cannot enable successful recognition, see Section 3.3.4. This is also the reason why it does not make sense to include different sized elements in a marker, because the smallest element determines decodability. For those reasons, a square layout was chosen for the markers, with each marker made up of n^2 square elements.

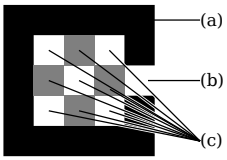


Figure 5.10: Schematic display of a single marker with labeled components. The border (a) allows recognition while the hole in the side (b) fixates the orientation. Those parts are always identical, independent of the marker id. The payload for the actual marker identification (c) is denoted with a gray-white checkerboard pattern and is set to a black and white binary pattern, according to the marker id.

For the most part the marker structure follows previous approaches, with a black border surrounding the identification pattern on the inside of the marker, see Fig. 5.10. Compared to other systems, the orientation fixation is provided using a single white square in one side of the border. This provides orientation fixation without decreasing the number of possible identification patterns, increasing the number of unique markers compared to other methods, see Section 3.3.2.

5.6.1.2 Multi Marker Pattern

Placement of multiple markers may be achieved with two methods. Either markers are aligned on an regular grid. Spacings between markers enable recognition, an example of this are the `aruco` markers [GJSMCMJ14]. Alternatively, markers may be placed in a checkerboard pattern like the one used in `CALTag` [AHH10]. See Fig. 5.11 for this type of layout. This layout leads itself to a corner based marker detection, while separated markers can be processed using blob detection. Note that the saddle point corners in a checkerboard pattern provides better refinement targets than the edges of individual markers, see [AHH10].

5.6.1.3 Inverted Markers

Note that the provision that a marker is determined by a black border is quite arbitrary, and markers may as well use a white border on a black background. Indeed, with a checkerboard pattern it is possible to place such inverted markers inside the space left within the checkerboard pattern, see Fig. 5.12.

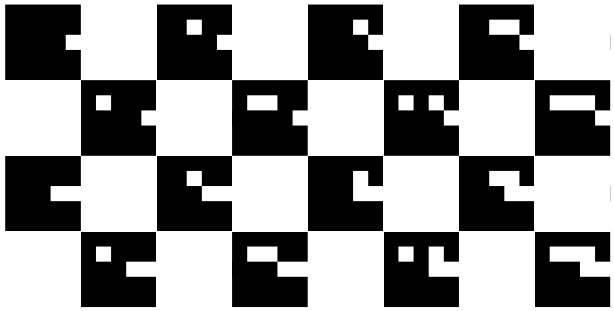


Figure 5.11: Figure showing the grid layout for multiple markers. Markers are arranged in a checkerboard pattern, while marker ids count up. Counting follows two consecutive lines in zigzag pattern. The codes in the first row are: 0, 2, 4, 8. The second row gives the values in between: 1, 3, 5, 7. The full grid has size of 32×32 markers, depicted here is a small crop from the top left corner. The third row therefore starts with 32, 34 ,... and the fourth with 33.

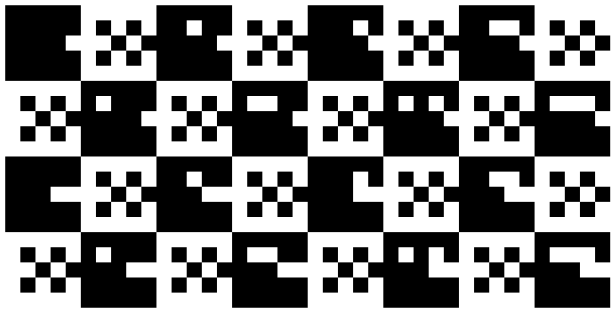


Figure 5.12: The same layout as shown in Fig. 5.11, but with the page id added to the grid. As the pattern codes a single page the page id is the same for all visible markers.

By combining regular and inverted markers for the marker identification, it is possible to double the number of bits available for encoding the marker id. However this also means that for unambiguous identification, at least one regular marker and one inverted marker have to be recognized.

The specific implementation chosen to provide these combined markers is based on the concept of pages. One page is defined as a checkerboard of markers, where the regular markers count up through the available range of ids, while the inverted markers determine the page number, with a fixed value per page, Fig. 5.12 demonstrates this approach.

Listing 5.1 This function encodes the marker id, by XOR'ing a different value for every two consecutive ids. This is a very simple scheme, a better approach would probably use some pseudo random bijective mapping to provide a less regular structure.

```
int calc_marker_code(int id)
{
  if ((id&2==2))
    return id ^ 170;
  else
    return id ^ 340;
}
```

Listing 5.2 The function shown here is used for deriving the page code using an id and the offset to that id. Id and offset are decoded to an position on the multi marker layout stored in j and i, the result is processed with a simple hash code. Note that the same page marker may be accessed trough any neighboring marker, the position is coded using the absolute position on the grid, therefore any correctly decoded marker may check any page marker on the same page for its respective page number.

```
int calc_page_code(int page, int id, int x, int y)
{
  int j = (id / 32) * 2 + (id % 2) + y;
  int i = (id % 32) + x;

  return (j*13 + i*7) % 512;
}
```

5.6.1.4 Irregularization and Error Detection

A marker grid as described above is very regular, see Fig. 5.12, which results in uneven distribution of black and white elements, problematic for the marker detection. Also the same page pattern is repeated for a full page, which means an image distortion like blur or nonlinear curve operation can corrupt all page ids in the same way, a problem to be aware of in the absence of bit error correction. Neighboring markers are also similar enough, that corruption may lead to the same bit being flipped for a whole group of markers. Therefore the markers are encoded with the code shown in Section 5.6.1.4.

Page ids are coded depending on their position in the grid, see Listing 5.2. This position is known on decoding because the marker id is decoded first, see Section 5.6.2.7. Apart from making the inverted patterns more irregular, this also serves as part of the error detection, as wrong decoding of the marker id leads to wrong decoding of the page id, and such errors can be detected by comparing neighboring markers. The result of these codes is a grid as shown in Fig. 5.13, which shows the same ids as Fig. 5.11, with the only difference being the application of the irregularization scheme.

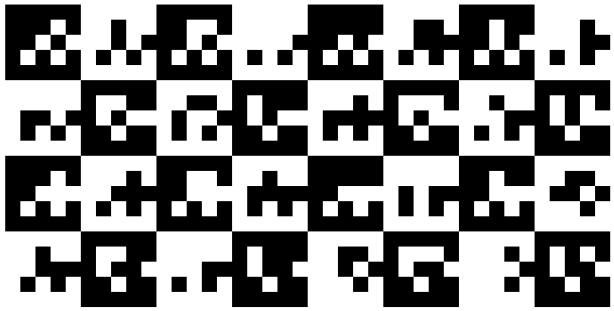


Figure 5.13: Once again the same markers as shown in Fig. 5.11 and Fig. 5.12, but with the irregularization scheme applied. Note that the pattern still encodes the same page id for all markers, but this time the page markers are modified according to the position in the grid.

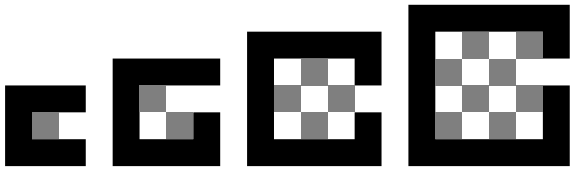


Figure 5.14: Possible marker sizes and the effect on the number of bits usable for identification. The grayscale checkerboard pattern denotes the bits which may code the id. Effectively usable bits are doubled due to the paging scheme. Still, a marker of 5×5 elements is the first to give a useful number of distinct markers at 262144. If printed at a size of 1 cm for the individual markers, the full addressable marker grid with would cover an area of $5\,\text{m} \times 5\,\text{m}$.

5.6.1.5 Marker Size

The number of distinct markers depends on the size of the markers. For a square marker of size $s \times s$ elements, the number of bits left for identification after subtracting the border is $(s-1)^2$, which is also the number of pages which can be addressed using the inverted markers. Therefore the effective number of markers m_e, that can be identified by the system is:

$$m_e = 2^{2(s-2)^2} \tag{5.1}$$

Fig. 5.14 shows different sized markers. Note that Eq. (5.1) is a very steeply increasing function, a size of $s = 4$ only offers 256 distinct markers while for the next larger marker size of $s = 5$, Eq. (5.1) gives 262144 distinct markers which should be enough for most purposes and therefore is the size chosen in this implementation.

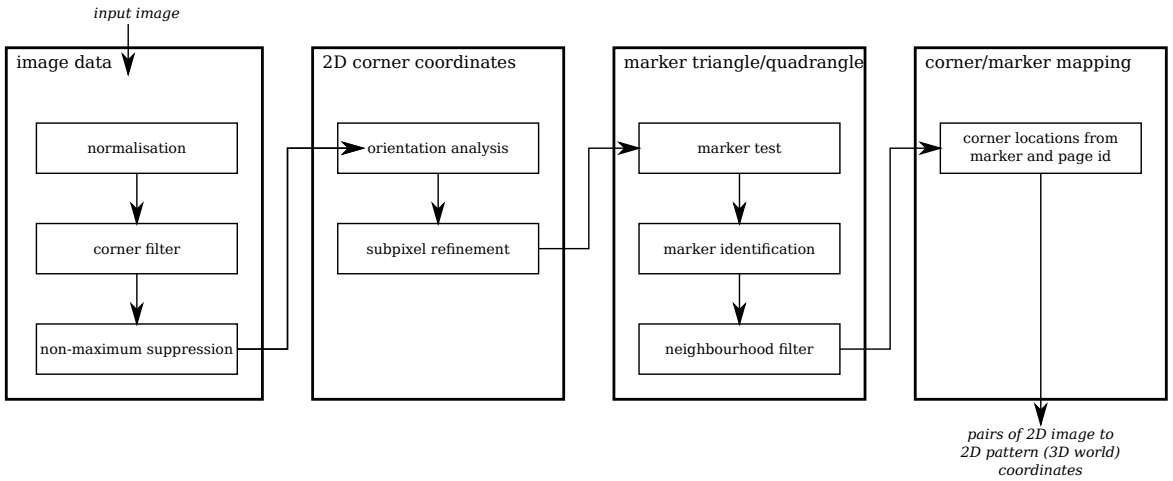


Figure 5.15: The stages of the marker detection procedure, grouped according to the type of information processed. Corner detection is run on bitmap data, while orientation analysis and subpixel refinement work with floating point subpixel accurate coordinates, although they still use the original image data for refinement. Three corners are combined to a marker candidate and candidates positively recognized as a markers are refined with a fourth corner for identification. In the end the system calculates the corner positions on the marker grid and outputs this position together with the respective image coordinates to be used by a PnP solver to perform camera calibration.

5.6.2 Detection and Identification

For an overview of the whole procedure see Fig. 5.15. First the image is normalized, then corners are detected using a simple filter. Corner orientation is estimated and corners are refined to subpixel accuracy. Afterwards markers are derived by testing all corner combinations and the identified markers are filtered, removing outliers.

The whole process is tuned for small markers, large markers are detected using a scale space approach starting with an image scaled down by a configurable factor, repeating the whole detection process at every power of two scale, until the full resolution is reached. For scaling the `resize()` function from `OpenCV` is used, with the `INTER_AREA` interpolation mode. This approach also decreases processing times as marker detection does not have to consider far off corners, and increases accuracy as lower scales exhibit less noise and other image distortions. Markers with a high confidence also block the area covered by them from being processed at higher scales, increasing the performance.

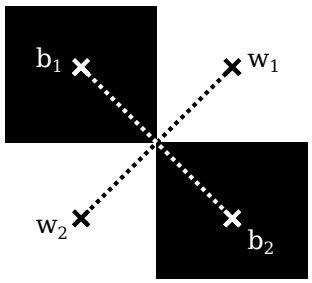


Figure 5.16: The corner score for a fixed orientation is derived from four samples. Two samples opposing the corner and two samples derived by rotating the first two by 90° around the corner.

5.6.2.1 Local Histogram Normalization

As images may be taken under differing illumination, and illumination may also be uneven within a scene, it is necessary to normalize the input image, so that further processing steps have a reference for what represents black and white in the image. Adaptive threshold methods do not keep the tonal information from an image and introduce typical staircase artifacts, resulting in the loss of subpixel information. To avoid such problems, the normalization uses a per pixel histogram and stretches the pixel value under the assumption, that the first 8-quantile from the histogram represents black and the last 8-quantile represents white. Because a per pixel histogram can be implemented using a moving window method performance is adequate, comparable to a median filter with the same range. Effectively this method provides the robustness of an adaptive threshold using the median, but without loosing the tonal information, useful for subpixel edge and corner detection.

5.6.2.2 Corner Detection

A generic corner detector can be used to detect the corners from the normalized input image, but tends to give false positives for the insides of markers, as the inside contains corners which a generic detector should detect, but which are not checkerboard corners. Also a tailored checkerboard detection gives better localization results, see [BL14]. The following method is based on a few observation about corners in a checkerboard: Corners in a checkerboard are symmetric, which means a value observed in one direction from the corner can also be expected in the opposite direction. Turning the checkerboard around the corner by 90 degrees inverts the corner, so that black parts come to lie on white parts and vice versa. The checkerboard corner detection exploits this rotational inversion by using the sum of two directly opposing samples and calculating the difference between this sum and the sum of the two samples that are positioned 90 degrees around the prospective corner, see Fig. 5.16. An additional regularization term is introduced to penalize the difference between the opposing pixels, exploiting the property of symmetry of checkerboard corners. The formula for the corner score

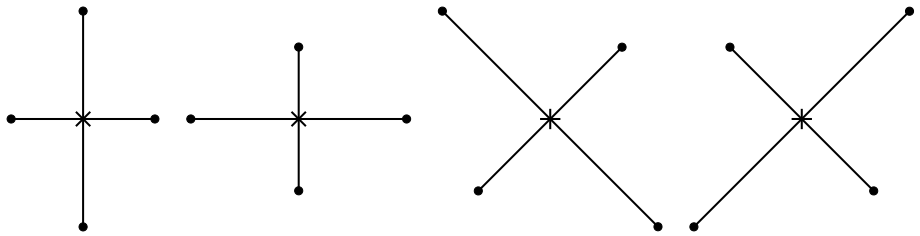


Figure 5.17: The corner score is calculated in four different configurations, for the final score the maximum is taken. In the image the respective sample positions are indicated with a dot, with lines connecting the opposing samples over the corner. This arrangement allows recognition for rotated and asymmetrically stretched markers, as obtained by an image taken at a low angle.

s_c is given in Eq. (5.3), where the w_i are two samples opposing the corner and b_i are the samples rotated by 90 degrees around the corner, compare Fig. 5.16. Note the similarity of this formula to the one used in [BL14].

$$s_c = |(w_1 + w_2) - (b_1 + b_2)| - 2|w_1 - w_2| - 2|b_1 - b_2| \tag{5.2}$$

As the input image may represent the checkerboard pattern rotated as well as asymmetrically stretched, due to perspective, the score is calculated with different corner sample distances for the w_i and b_i, as well as with several rotations, and the maximum over all iterations is used for the final score. Fig. 5.17 visualizes the used parameters. The maximum is used because corners are assumed to have only few pixels available for detection, hence incorrect directions result in arbitrary results, depending on the surroundings. Compared to that Bennett and Lasenby use a summation [BL14], more useful if the whole area incorporated in the score is part of the corner. For robustness against noise and aliasing, the input image is filtered with a Gaussian filter with low strength, prior to the corner detection. To derive suitable corners from the image filtered by the corner detection, a non-maximum suppression step is added at the end and all pixels above a configurable threshold which are maximal within a 3×3 neighborhood are selected as corner candidates.

5.6.2.3 2D Orientation Analysis

For the subsequent processing steps it is necessary to estimate the two directions of borders that make up the corners. The second boarder may not be orthogonal to the first one due to perspective. To estimate the direction several orientations for the first direction are tried, with the second direction fixed at a 90° degree offset. The direction obtaining the maximum score is selected and then both directions are iteratively improved from the initial guess until no improvement can be made. The score is calculated by reprojecting a small patch from the image, centered around the corner, and using an affine transform derived from the two orientations vectors. The transform uses bilinear interpolation to allow subpixel precision.

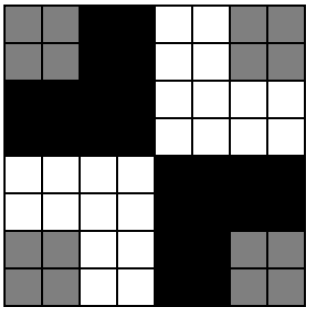


Figure 5.18: The template used for corner estimation. The white areas denote pixels which are summed up to the white average, black likewise for the black average and gray pixels denote the region which is not incorporated into the final score.

From the reprojected patch the score is calculated using a simple template matching method, where the template is a perfect checkerboard corner, see Fig. 5.18. The average for what should be black and white samples in the patch are calculated according to the template and the corner score is calculated as the difference between the averages using Eq. (5.3):

$$\text{score} = |\text{avg}_{white} - \text{avg}_{black}| \tag{5.3}$$

The iterative optimization starts with a large step, iterating until no improvement is possible, and then restarts with a smaller step up to a configurable smallest step.

5.6.2.4 Subpixel Refinement

To refine a corner with subpixel accuracy, the same method is used as for the orientation analysis, by shifting the corner center by a fraction of a pixel, refining orientation and restarting from the new location if the score could be improved. Different directions are tried in turn until no improvement can be made. The refinement works analogous to the orientation analysis, by first starting with a large step and then repeating with successively smaller ones, up to a configurable minimum step size, which defaults to 0.04 pixels. Subpixel refinement and orientation analysis result in a final score for the quality of a corner, which is used to filter out unlikely candidates using a fixed threshold.

5.6.2.5 Marker Test

The assembly of candidate corners to discrete markers is a simple brute force method. Basically all combinations of corners within a range are scored to find the combination with the highest score. To reduce the performance impact, corner combinations are first checked for a range of conditions, which mark them as invalid before any expensive calculations have to be made, see Section 5.6.2.6. Only three corners are used to calculate the score, resulting in an affine

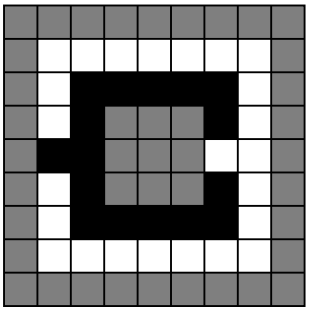


Figure 5.19: The template used for marker detection. Again the white areas denote pixels which are summed up to the white average, the black areas are used for the black average and gray pixels denote the region which is not incorporated into the final score.

transform. For identification a fourth corner is used for a correct projective transformation. This speeds up recognition, as less combinations have to be checked, without negatively affecting the identification which can work with the correct projection.

The score is calculate using black and white averages for a marker template based on the black and white of the border of the marker and the area surrounding that boarder, see Fig. 5.19. The result is used to normalize the reprojected marker. This is done in addition to the normalization at the beginning, because now the samples with the exactly known reference values can be used for normalization. The marker score s_m is then calculated using the average for the black and white border, here b_{avg} and w_{avg}, and with the difference between the hole w_h in the marker border and the hole b_h of the inverted marker, directly to the left. The full formula is given in Eq. (5.4):

$$s_m = \frac{w_{avg}}{b_{avg}}(w_h - b_h) \tag{5.4}$$

The extra term with $(h_w - h_b)$ is necessary to provide correct orientation, giving a high weight to the samples responsible for orientation fixation.

5.6.2.6 Fast Rejection Checks

Because candidate corners already possess orientation information, the orientation of candidate markers can be compared to the orientation of the corners, if the difference is above a threshold the marker is rejected without expensive score calculations. The other checks performed for fast rejection are: Marker size within an allowed range, polygon direction, as wrong direction would result in a mirrored marker, and a pre-score with a metric identical to the one used in the corner filter, but using the direction and distance given by the marker and without the regularization term.

5.6.2.7 Marker Identification

Once a combination of three corners results in a score indicating a likely marker, the fourth corner is estimated from the positions of the first three and then refined using the refinement method above. The final score and identification is then calculated from the perspective projection derived from those four corners, using simple thresholding to obtain the original bit values.

Reprojection uses simple linear interpolation and therefore introduces a certain blur. Combined with possibly noisy and blurred input, this results in reprojected markers which may not be as clear as necessary for correct identification. For example a single white pixel surrounded by black may be pushed under the threshold due to blur. Experiments have shown that a simple sharpening filter increases the identification rate, even if the input does not obtain any blur to begin with.

To retrieve the full 18 bit marker id the neighboring inverse page markers also have to be identified. The four neighboring page markers are all estimated from the corner positions of the marker in question, refined and identified as above, yielding up to four possible page numbers. The final marker page is then calculated by weighting the different page number with the scores of the respective inverted page markers. The highest score is selected as the page id.

5.6.2.8 Neighborhood Filter

As there is no error correction incorporated into the markers per se, errors in the corner refinement, noise, as well as random image structures and occlusions, can lead to the erroneously identification of markers. To filter out those anomalies, markers are compared with their neighbors at the end. A correctly identified marker needs a configurable amount of neighbors at roughly the expected position or else is considered invalid. This means error detection is performed using the knowledge of the layout of the marker pattern at no additional cost in terms of bits used. This is an advantage compared to single markers using error correction as in 3.3, especially as the number of neighbors required for correct identification can be adjusted at the time of identification and therefore the trade off between correctness and sensitivity of the detection may be changed at will. On the other hand, errors are impossible to correct, only error detection is possible.

5.6.2.9 Output

The final output of the whole process are not the individual markers, but the detected corners, along with their ids, as position on the marker grid. This simplifies work with marker grids, as corners are explicitly addressed by their position, making the derivation of the 3D position trivial. Also each corner is part of two markers, which means that failure to detect single individual markers has no direct consequence on the output of the detection process.

5.7 Software

The remaining software written for this work, excluding compression and marker detection, connects the different parts of the system, implementing the recording in Section 5.7.2, to a simple file format described in Section 5.7.1, which makes use of the `BBP` compressor. For evaluation, a range of tools have been written and are introduced in Section 5.7.6, as well as an application which allows interactive rendering from the light field dataset, see Section 5.7.4.

The implementations use several open source libraries, notably `OpenCV` [CAP+12] for image processing and camera calibration, and `pcl` [RC11] for point indexing and search.

5.7.1 File Format

The implemented file format is a very simple ad hoc format, designed from the need to minimize processing and delays while incorporating the required compression into the format. Nevertheless the format is extremely simple, consisting of a header with a fixed size of 4 KiB which starts with a magic number and contains a number of fixed fields used to describe the file as well as two offsets pointing to the index, consisting of two arrays, which contain offsets of individual frames their size respectively. Most of the header is zeroed out, allowing later extension. Writes to a file only append data. To sync the file index, the two arrays are appended to the file and the header is updated to point to the new position. Any write of either a frame or the index are executed asynchronous, and only block if I/O cannot keep up. The file format is implemented in a small c library named `lfio`, providing the asynchronous read and write functionality.

The I/O library also includes the compression functionality, using one of three modes. The first is the fast `BBP` coder, developed in this work, see Section 5.5. The second mode uses one of the video coders available in the `ffmpeg` [FFM] library, with prior lossless format conversion, see Section 2.4.2.2. And third, any of the compression formats available through the `squash` library [SQU] can be used, which includes formats like `gzip`, `bzip2` or `lzo`. By convention the filename extension is lff for light field data and lfi for auxiliary information, like calibrated camera positions.

5.7.2 Recording

The nature of the `lfio` library enables asynchronous I/O for recording, avoiding delays due to disk I/O. The recording tool **lfrecord** is similarly implemented with focus on low processing needs, even avoiding memcpy by passing around preallocated buffers. The recording software is only compatible with XIMEA cameras implementing the `XiAPI` [xim03], and also monitors the timestamps attached to individual frames by the camera, registering dropped frames. Additionally the camera temperature is monitored and saved.

5.7.3 Calibration

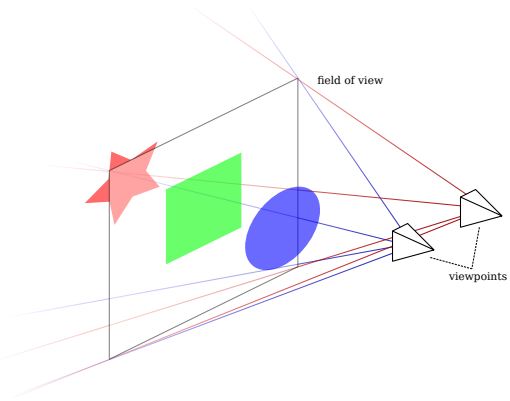
The calibration tool **lfcalibrate** uses the high density marker system introduced in Section 5.6. The tool implements the intrinsic calibration of the camera and the extrinsic calibration for all frames. Calibration uses the respective functions from the `OpenCV` library: `calibrateCamera` for combined intrinsic and extrinsic calibration and `solvePnPRansac` for extrinsic calibration. Because `calibrateCamera` cannot detect outliers, points are always filtered using `solvePnPRansac` with either a previously calibrated model or using a non-corrected camera model with a large threshold to provide robust estimation. The `OpenCV` camera calibration is based on the method of Zhang [Zha00].

5.7.4 Rendering

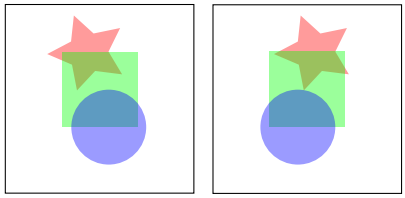
The rendering methods, available in the form of a the **lfavg** tool, are provided mainly as a means for demonstration and verification of the dataset, therefore only few features were implemented. Specifically synthetic aperture rendering and a simple resolution enhancement. The rendering is always executed in parallel for several frames, the asynchronous nature of the `lfio` library is exploited by requesting a number of frames, and after a short break, default 50 ms, the available frames are rendered while the remaining frames are read in background. This allows interactive execution, as the currently rendered output is always shown. Out of focus regions appear first in low quality, with double images, but quickly improve, as more frames become integrated into the result. Magnification, as well as position and rotation of the focal plane can be freely chosen. The viewpoint may only be rotated around the scene, as viewpoint movement towards the scene center would need a more complex rendering algorithm than the implemented perspective projection, see Fig. 5.20.

5.7.4.1 Synthetic Aperture Rendering

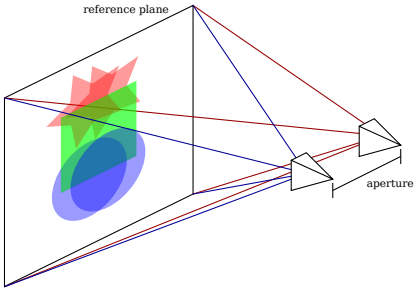
Synthetic aperture rendering simulates an aperture by taking images from viewpoints placed within the simulated aperture. Rendering takes place by perspective reprojection and averaging of all reprojected images. The focal plane is simulated by intersecting the corners of a virtual



(a) A scene consisting of three shapes in front of each other, recorded from two different viewpoints.



(b) The recorded views differ, with disparity depending on depth of the respective feature.



(c) Reprojection of different views onto a reference surface.

Figure 5.20: Visualization of focus blur in synthetic aperture rendering. In Fig. 5.20a a scene is captured at multiple viewpoints, resulting in different placement of objects depending on the distance from the viewpoint, see Fig. 5.20b. A synthetic aperture rendering can be produced from multiple viewpoints, if viewpoint extrinsics and camera intrinsics are known, by intersecting the views with a reference plane, and reprojection of the observed images onto this plane using the four corner points, which results in a perspective projection, see Fig. 5.20c. The result is that features originally lying on the reference plane are projected onto that same position by all views and remain sharp, while features in front and behind get projected to different positions, depending on the depth and the viewpoint. The synthetic aperture is constructed from the viewpoint positions. Two views give strong double images instead of defocus, but more views or good viewpoint interpolation can give a smooth appearance.

camera with the desired focal plane and then reprojecting all images so that the respective corners from the virtual camera, in each viewpoint, are projected onto the same four corner points of the reference plane, see Fig. 5.20 for a visualization. Images are combined using the arithmetic mean. Per viewpoint weights could be used to alter the appearance of the aperture. Projections are all calculated using the calibrated camera parameters, and image reprojection is performed using the `warpPerspective` function from `OpenCV`. Reprojection uses bilinear interpolation, with prior demosaicing of the raw Bayer images. Focus blur becomes apparent for objects which do no lie on the focus plane, as those are projected onto different positions in the averaged image, depending on the viewpoint of the respective camera, see Fig. 5.20. The implementation allows modulation of depth of field, by allowing free movement of the focal plane, including rotation, and by allowing the selection of an aperture size. See Section 6.3.1 the results obtained with this method.

5.7.4.2 Enhanced Resolution Rendering

The enhanced resolution rendering, implemented in this work, replaces the bilinear interpolation as used by `OpenCV`, with Shepard's method, which implements an inverse distance weighting with an exponent of 2, in an subpixel accurate forward projecting scheme. For points on the focal plane, the full resolution of the light field is reproduced at the rendered image resolution. In such a reprojection noise can be nearly completely eliminated, thanks to the averaging of a large number of images. Compared to the bilinear backward projection approach in Section 5.7.4.1, the resolution is increased mainly because Bayer pattern demosaicing is avoided, by projecting individual color pixels. The resultant image is well suited for ordinary 2D image sharpening as it is very clean, resulting in an subjectively higher resolution image, see results in Section 6.3.3. Note that this enhancement is only possible for objects on the focal plane, as for objects placed in front or behind the focal plane the individual reprojected samples are spread over a larger area. Out of focus areas of the regular reprojection appear as a uniform blur, as they are the result of an arithmetic average over all individually interpolated images. But for the enhanced resolution rendering those areas are a result of a weighted average, based on the subpixel positions on the reference plane. For areas in front of the focal plane the effect is similar to the normal synthetic aperture rendering, but for areas behind the focal plane the direction on the image plane is reversed, as samples, for example from "left" and "right" viewpoints cross at the focal plane. Because subpixel positions are still defining the weighting of the samples, the image is constructed of small inverted parts, with discontinuities when the weight flips from one pixel to the next. The enhanced resolution rendering mode also more readily display aliasing in the dataset, because the full resolution of the dataset is exposed, where the regular reprojection induces additional blur due to interpolation and demosaicing.

5.7.5 Motor Control

The motor control uses a STMicroelectronics L298N dual H-bridge motor driver, which is controlled via the GPIO ports of a Raspberry Pi [RBP03] single board computer. To enable PWM control of the motors for smooth acceleration and braking, as well as speed control, the DMA engine is exploited, allowing PWM control of arbitrary GPIOs with a precision of 1 µs (1 MHz). The code for GPIO control using DMA was taken from the `RPIO` [RPI03] library. The implemented solution allows smooth motor control and is controlled via network. The setup includes three contact sensors, two to detect the lower and upper limit of the robotized arm, and one to detect the completion of a single rotation by the turntable. After each rotation of the turntable the arm is moved by a small amount, until either the program is stopped or the upper limit of the arm is reached.

5.7.6 Auxiliary Tools

This work required the implementation of several additional tools for handling and evaluation of the used methods and recorded data. For the core tools a more detailed description, as well as options and usage, is available in Appendix A.1.

lfconvert allows the conversion between different compression formats, useful for recompression after capture, for extraction of parts of the dataset, and for evaluation of compression methods.

lfcmp compares two lff files and may be used to verify the lossless property of the used compression, or to calculate PSNR scores for lossy compression methods.

lfbrowser allows playback of lff files as a video stream, including interactive control like pause and rewind.

lfextract allows extraction of frames from an lff file, together with the corresponding extrinsics from an lfi file, meant for input to other tools, for example for visualization.

lfstreamsnr allows calculation of dynamic range by using a user selected black part from the frames of a lff file as dark frame, incorporating the calibrated non-linear response of the camera. It can also be used to calculate camera PSNR using a recording of a static scene.

lfinfo extracts the camera extrinsics from an lfi file and saves them in plaintext, useful for further processing, like visualization of viewpoint positions.

lfsearchsampling was meant as a tool for finding discrepancies in the dataset by comparing subsequent frames, and detecting regions of high change. However this simplified approach does not work for the dataset, as small rotations of the camera possibly caused by vibrations in the frame cause subsequent frames to not point in exactly the same direction, which is

not a problem for the rest of the system as such variations are compensated with the camera calibration, but they cause false positives for this tool, making it only useful to detect those vibrations, as well as skipped frames.

lfconv_bench is variant of the lfconvert tool which measures execution times and compression ratio for pairs of compression method and Bayer packing method, see Section 2.4.2.2. This is used to determine the best compressing packing mode for the compression methods.

lfconv_bench_single similarly benchmarks different compression methods, but with a fixed packing method per compressor, previously determined using `lfconv_bench`

lfconv_bench_gop also executes a benchmark for different compression methods, but varies the group of picture size, which determines the number of consecutive frames a coder can access to exploit temporal correlation.

6 Results and Limitations

This chapter presents the results which were achieved with the implemented design and provides comparative evaluations, where applicable. Limitations of the system are pointed out, which may serve as a starting point for improvements in a future designs. Possible future works, expanding on these findings are introduced in Chapter 7

6.1 Compression

This section provides an evaluation of the `BBP` compression scheme and a comparison with the fast lossless methods introduced in Section 3.2, as well as an evaluation of the high efficiency methods for optional archival of the dataset, see Section 6.1.3. Section 6.1.1 details the method of evaluation and Section 6.1.2 presents the results.

6.1.1 Evaluation Method

All performance values referenced here were measured on an Intel® Core™ i7-860 Processor at 2.8 GHz, using a single core. The evaluated video compression schemes were called trough the `ffmpeg` library, which also provides most of the implementations, aside from `x264` [X26] as an AVC encoder, `x265` [X2617] for HEVC and `openjpeg` [OPE] for `jpeg2000`. The remaining compression methods were called trough the `squash` library, which uses the respective reference implementations, listed in Section 3.2. The video coders use the respective pixel format which achieves the best compression ratio, see Section 2.4.2.2. For most of the coders the best compression was achieved using the rotation scheme of Zhang et al. [ZW06], only `jpeg2000` performed better with the raw Bayer pattern. The `ffvhuff` coder performed best with the individual color channels of the Bayer pattern placed side by side, and the `jpeg-ls` implementation of `ffmpeg` was not compatible with the color subsampling required for the rotation scheme. From the remaining formats the side by side format performed best.

Evaluation was performed on a short representative slice from the light field with a length of 32 frames. This length was sufficient for the tested methods to obtain near optimal compression performance, see Fig. 6.1.

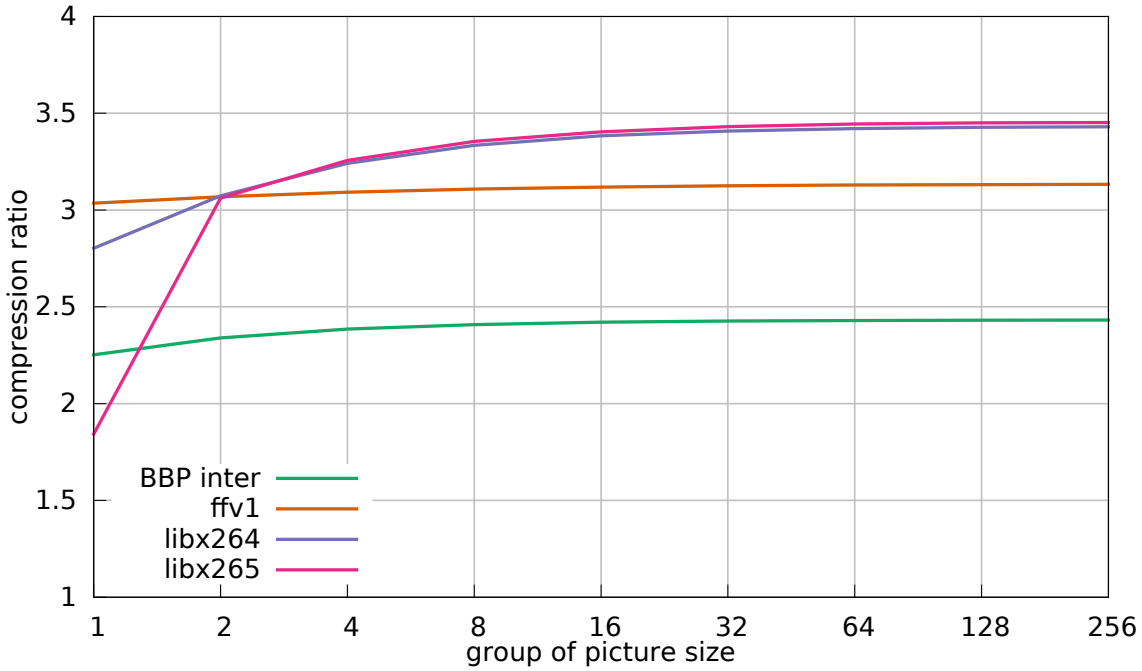


Figure 6.1: Plot showing the relation between the group of picture size, short GOP, and the compression ratio. The GOP size limits the range of frames, which an encoder can use for temporal correlation. It is apparent that for more than 32 frames the increase in compression performance is only marginal.

6.1.2 Compression Performance

The `BBP` method implemented for this work is the fastest by a wide margin, with a bandwidth more than 7 times that of the closest contender, see Fig. 6.2. On the other hand, compression efficiency is worse than nearly all dedicated image/video compression methods. Only `ffvhuff` is matched by the inter frame variant, at a block size of 8 Bytes. In summary, the envelope for image compression is extended towards a higher maximum bandwidth at the expense of compression efficiency. The implemented methods make generic compression methods superfluous for image and video compression, while those methods were previously slightly faster than the dedicated image and video compression methods. Note that the implemented method is the only one capable of exceeding USB3 bandwidth on a single core, making it very suitable for high bandwidth lossless video capture.

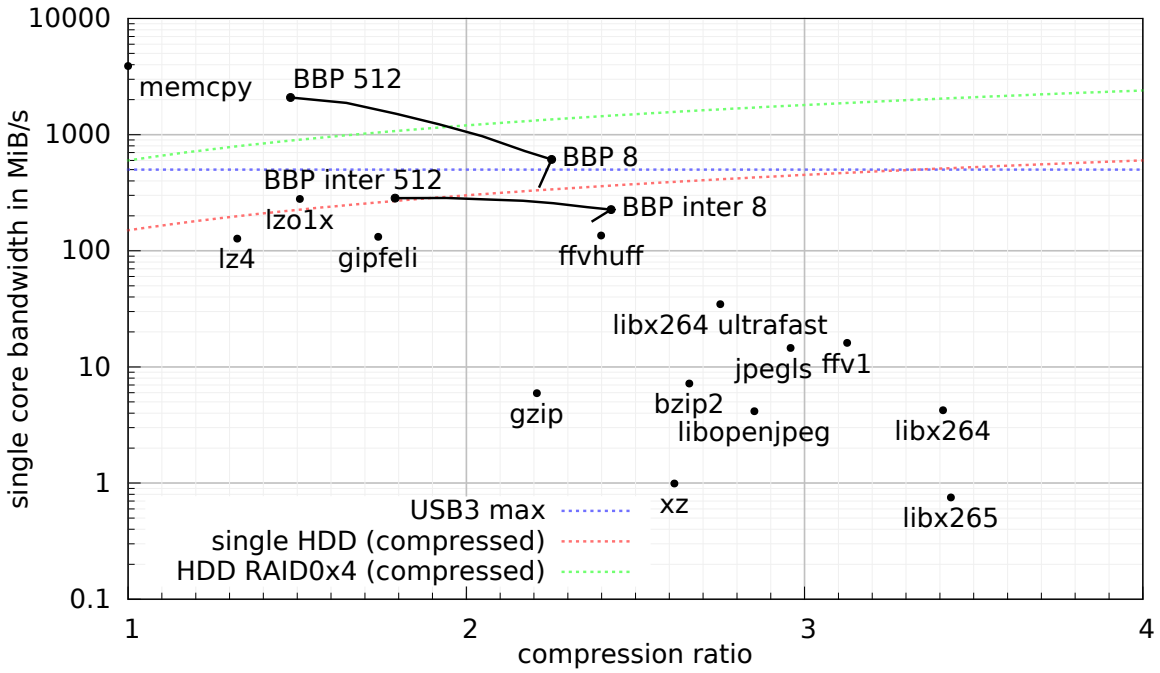


Figure 6.2: Single core performance of several lossless compressors. From the generic compression methods many were left out in this plot, due to similar speed and ratio, see Fig. 3.1 for the full list. The methods implemented in this work are denoted as BBP for the regular 2D method and BBP inter for the expanded 3D version. These methods outperforms all others, speed wise, and expands the envelope of image/video compression towards faster compression speeds, at a reduced compression ratio compared to other image/video compression methods.

6.1.3 Archival

While the measurements in Fig. 6.2 show that BBP is well suited for high speed processing, it is sometimes useful to achieve maximum compression with an offline method. Therefore one of the slower methods may be used to further reduce the size of the dataset, for example for archival or transmission of the dataset. Fig. 6.3 shows estimated conversion times and file sizes for the recorded dataset, assuming perfect scaling and utilization of 4 CPU cores. Only the methods which lie on the frontier towards a high compression ratio are included in this estimate.

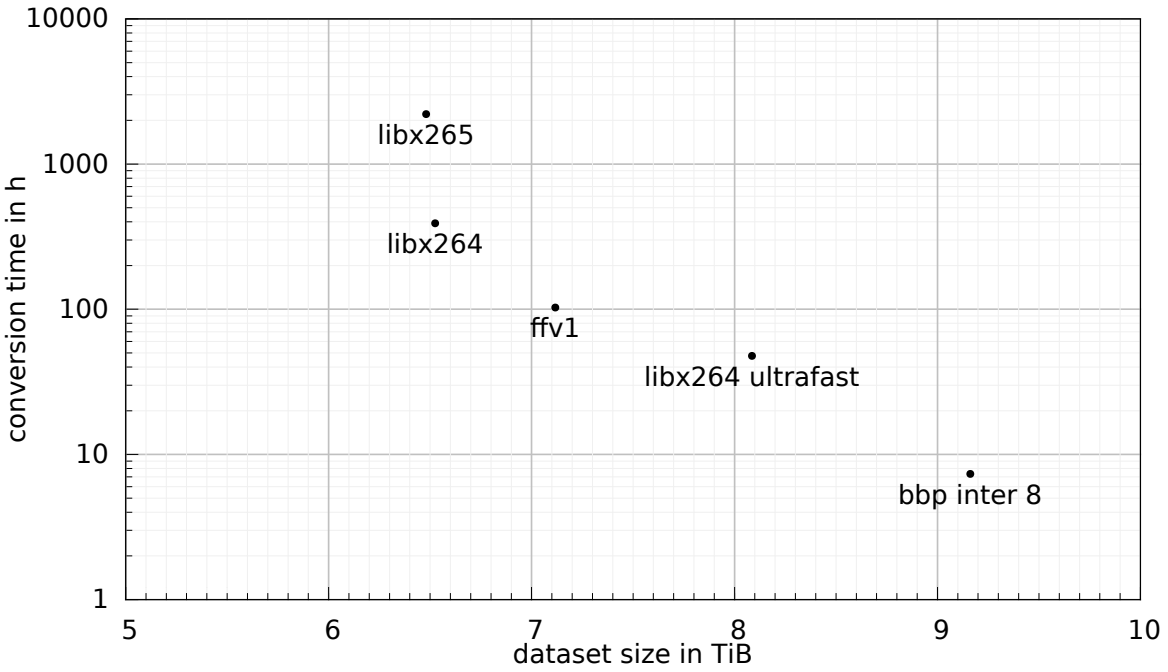


Figure 6.3: Plot showing the estimated conversion time for the dataset versus the final file size. Only methods with unique trade offs are included, `BBP` inter is included as a reference for the fast methods.

6.2 Calibration

This section present the result for the implemented `HDMarker` system for marker detection, introduced in Section 5.6, as well as the results when using this system for the calibration of the light field dataset.

(a) Input image with strong perspective distortion, note low resolution and the difficulty to make out the bit patterns for markers at the far end.

(b) Processed image annotated with correctly detected corners.

Figure 6.4: Demonstration of the detection performance with bad illumination and strong distortion. Note that markers are detected right up to the five pixel size limit. Markers smaller than that are impossible to identify as the 9bit identification code is placed in less than 9 pixels.

6.2.1 Marker Detection

The robust detection process is capable of reliable recognition and identification of very small markers under difficult illumination and under strong perspective distortion. Marker detection is feasible down to short side of 5 pixels, the absolute limit dictated by the size of the markers itself. Fig. 6.4 gives a good estimate as to the performance that can be achieved with this method. The test performed in Section 3.3.4, shows that `aruco` markers can be detected down

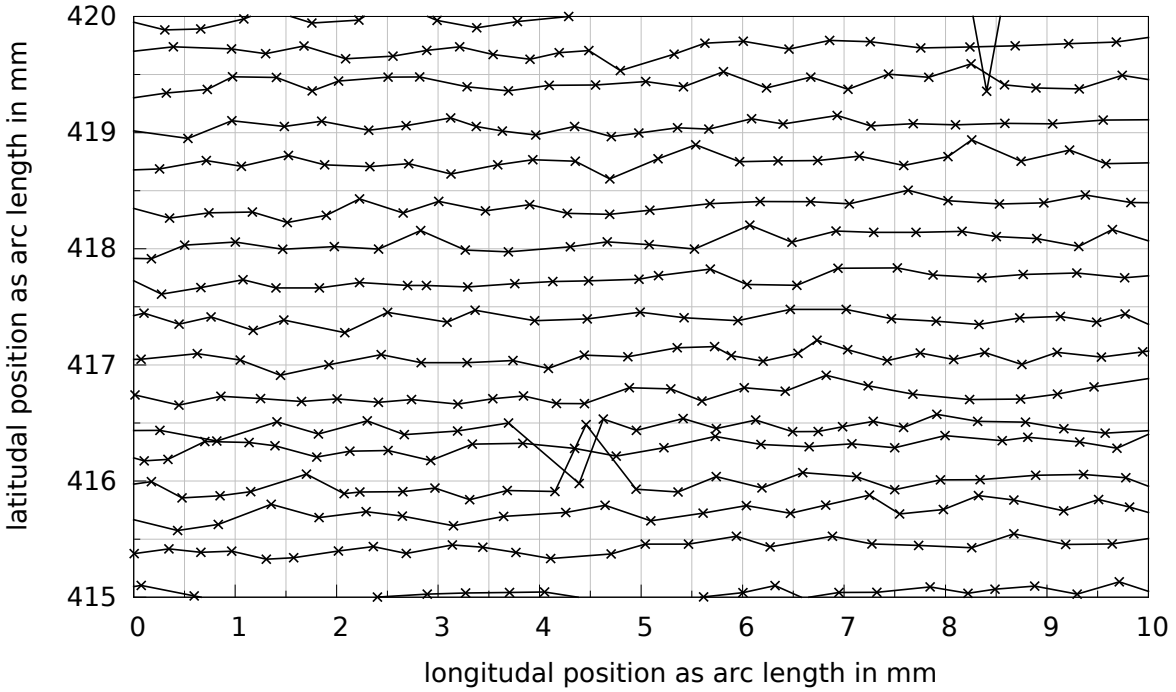


Figure 6.5: This plot shows calibrated marker positions from the recorded light field dataset. It is apparent that precision of the mechanical setup and the calibration is sufficient to resolve the path of the camera as it samples successive horizontal slices. Target viewpoint distance was 420 µm, a goal not always reached. Note a few outliers, possibly due to vibrations or uneven turntable movement.

to a size of around 10 pixels. This is exactly double the size required for `HDMarkers`. The system also provide a higher number of usable markers at 262144, compared to between 280 and 4146 for other systems, see Section 3.3 and [Ols11]. However, those advantages are bought at the expanse of significantly higher processing requirements: The processing of a single frame from the light field takes around 1 second, using all processor cores of the used processor and for an image size of 2048×2048, while previous methods are capable of realtime processing, although often at smaller image sizes.

6.2.2 Camera Calibration

Without an exact reference frame it is difficult to assess the full precision of the final calibration. As the reason for the use of fiducial markers was precisely to avoid the need for external measurements, such would be out of the scope for this work. However the consistency of the measurements may be observed. Fig. 6.5 shows a plot of calibrated viewpoints from the dataset. It is obvious that the calibration was able to resolve the path of individual slices well.

Note that no filtering of viewpoint positions was performed, marker detection and calibration were executed completely independent for each viewpoint. Also note that Fig. 6.5 includes the mechanical imprecision of the setup, a fact which may explain the few deviations in Fig. 6.5, e.g. from irregularities in the turntable. The precision of the calibration is also high enough to produce enhanced resolution renderings using subpixel accurate weighting, see Section 6.3.

Another way to assess the calibration performance is an error measure of the calibration itself. With several hundred detected reference points per viewpoint, the PnP problem, is highly over-defined, allowing the assessment of the performance using the root mean square of the reprojection error for the reference points, see Section 2.1.3. The resultant RMS error varies for different images of the dataset, but is within a range of 0.4 to 0.6 pixels.

6.2.3 Calibration Bias

A visualization of the reprojection error, as performed in Fig. 6.6, shows that there remains a localized bias in the error of the reference point positions. This means that the error is not the product of imprecision in the marker detection, as those would manifest as random directions, but rather the model used by the calibration is not completely accurate. The calibration model has the following shortcomings, though the magnitude of the respective inaccuracy is unknown:

Calibration target: The calibration target is a simple planar surface, constructed from a wooden plate. As the edge of the plate slightly protrudes the support structure below, the edge may slightly bend down due to gravity, and additionally the plate might not be perfectly plane to begin with. Accurate measurements of the target would be necessary to eliminate this problem.

Lens distortion: The distortion of the lens is modeled using a radial distortion model, including tangential distortion. However the distortion may actually be more complex, explaining the bias depending on the position within the image. A more complex distortion model, possibly using a model-less per pixel look up table, could improve the distortion correction.

Depth depended distortion: Image distortion may, additionally to radial position, also depend on the depth of features, see Alvarez at al. [AGS11]. As depth information is not directly available from 2D images, a solution would have to involve a multi-step procedure, using calibrated positions to supplement reference points with depth information, as an input to a depth based calibration step.

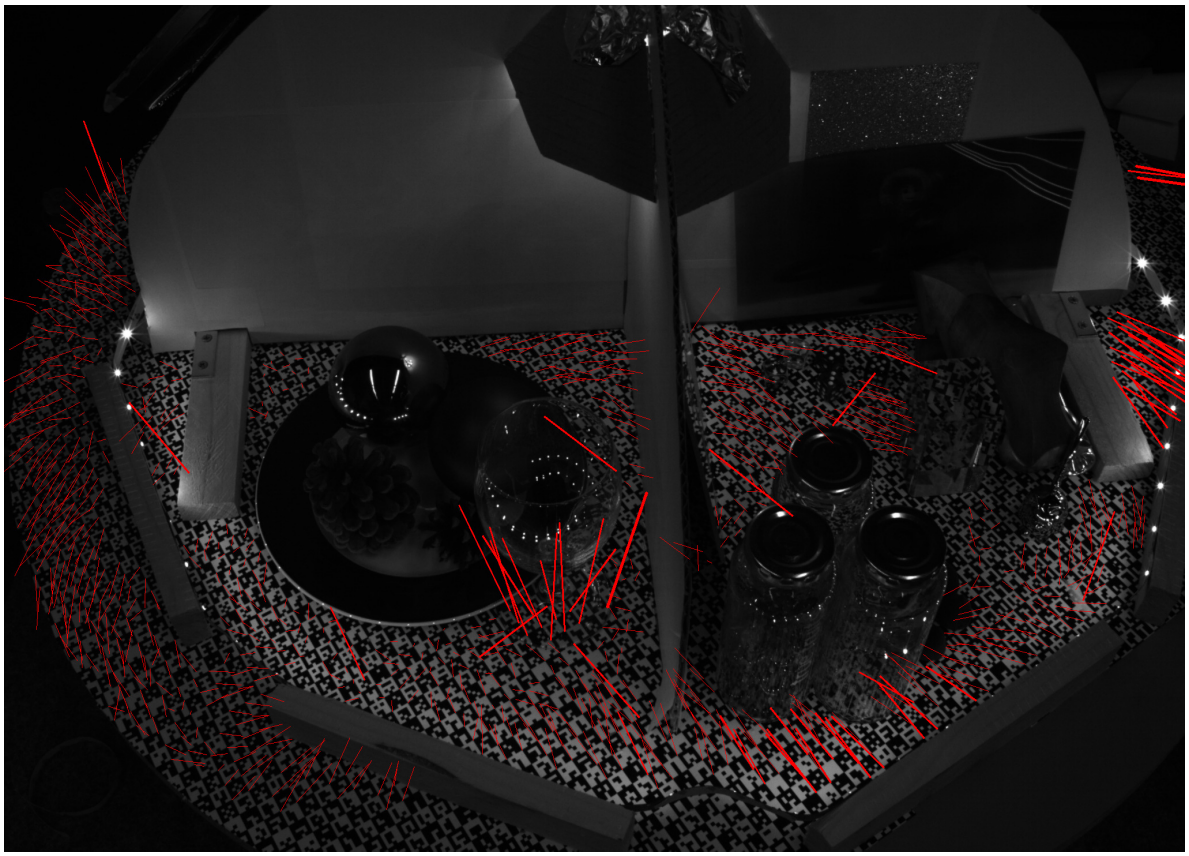


Figure 6.6: Visualization of the reprojection error, overlaid as red lines over the calibrated image. This image is not from the final dataset and was chosen because of the high number of visible markers, displaying the effect with more clarity. Lines are drawn from detected marker corners to the estimated position obtained by reprojecting the known 3D location. For visualization purposes the the lines are stretched by a factor of 100, e.g. a line with length of 100 pixels represents an error of 1 pixel in the original image. Thickness is also increased for errors over 1 pixel. From the image it is obvious that the reprojection error is mostly below 1, as most lines are thin. Also while most errors are relatively small at around 0.5 pixels, they are often orientated in the same direction, representing a bias, which indicates an non-optimal calibration model. Note that absolute reprojection error is still low, but the results could be further improved with a better model.

6.3 Light Field Dataset

Table 6.1 shows the properties of the recorded light field dataset. The recording took 18 hours, sampling slice by slice until the available storage capacity of 12 TiB was nearly exhausted. Due to the dense sampling the final dataset does not cover the full hemisphere, but only a small part thereof. The sampling is however dense enough to provide high quality synthetic aperture

number of viewpoints	5831235
total number of samples	$\geq 2.4 \cdot 10^{13}$
angular coverage of hemisphere	$360° \times 7.5°$
uncompressed size	22 TiB
compressed size	11 TiB
capture time	18 h
number of horizontal slices	431
average number of viewpoints per slices	13529
avg. horizontal viewpoint spacing	443 µm
avg. vertical viewpoint spacing (estimated)	302 µm
minimum camera scene distance	65 cm
scene diameter	70 cm
viewpoint resolution	2048 px × 2048 px
lens focal length	12.5 mm
recorded dynamic range	63 dB
saturated samples	0.73‰

Table 6.1: Parameters of the recorded light field.

rendering without viewpoint interpolation, see Section 6.3.1. Extrinsic camera calibration was also performed for the whole dataset. The precision of the calibration is sufficient to achieve increased resolution rendering, see Section 6.3.3, by using subpixel accurate weighting as described in Section 5.7.4.2. The recorded viewpoints form a band surrounding the hemisphere, with a distance from the scene center of 100 cm and with a height of 13 cm, allowing synthetic aperture renderings with a maximum aperture of 130 mm, if the aperture has to stay circular. With the focal length of 12.5 mm this results in an aperture of $f/0.096$, a value not currently possible with regular optics.

6.3.1 Synthetic Aperture Rendering

To evaluate the quality of the rendering provided in Fig. 6.7, an example of a real image, acquired with a regular camera with large aperture is available in Fig. 6.8. Several optical effects can be observed: The green patch in the background above the bottles and the giraffe

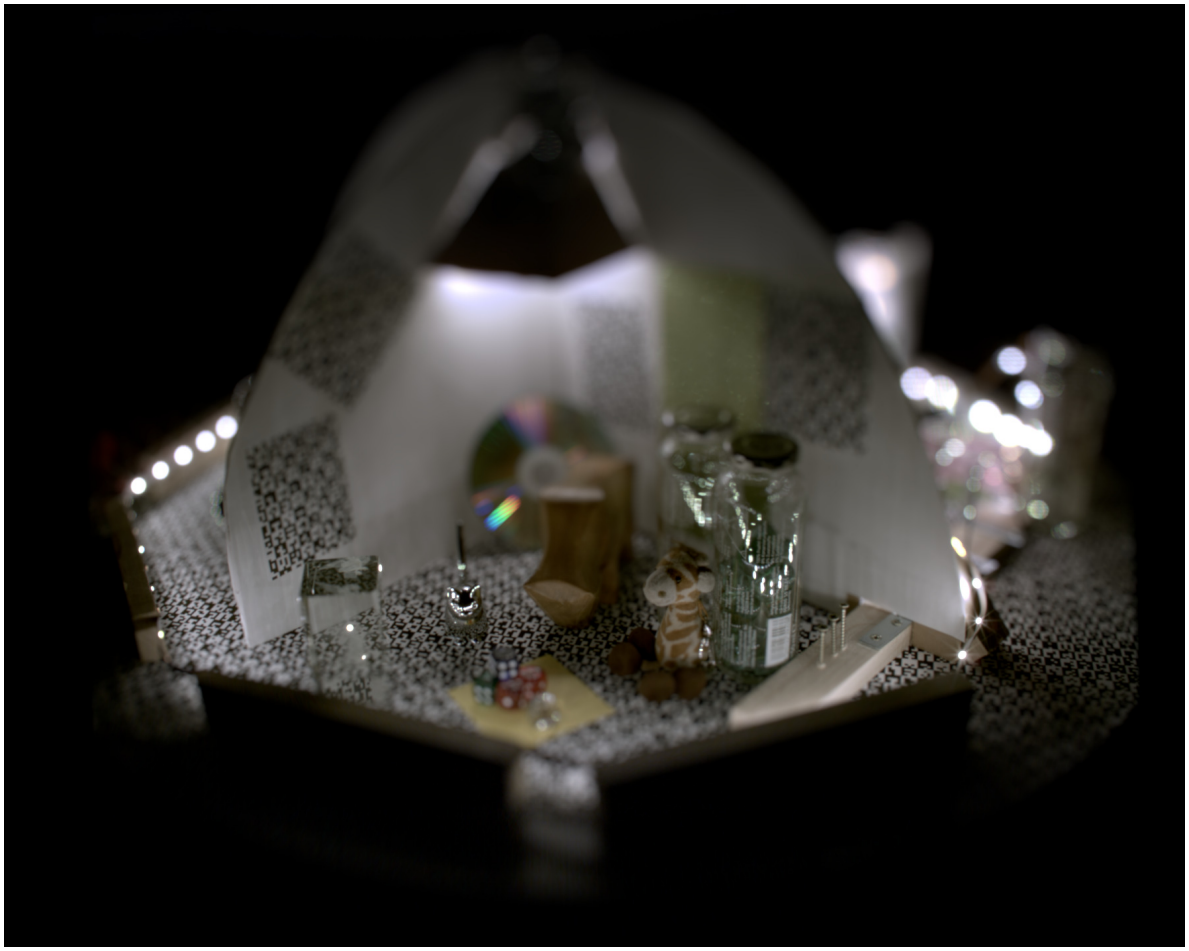


Figure 6.7: Synthetic aperture rendering, produced by integrating 4581 viewpoints, using the method in Section 5.7.4.1. Note the absence of double images and smooth background blur, even tough the image was produced without viewpoint interpolation. For the rendering the top of the focal plane was rotated toward the viewer, increasing the defocus effect.

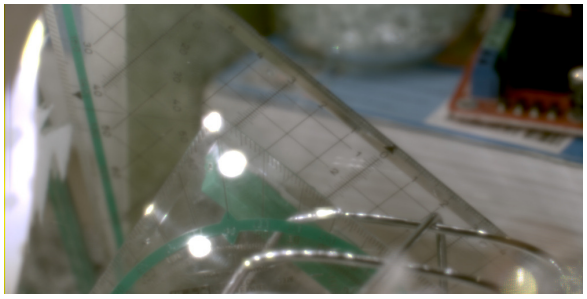
show small structures even though the area is out of focus. This is the effect of very directional highlights of the glittering surface. In a recording with less viewpoints some of the highlights will suddenly vanish, as the main contributors are skipped by the sampling. Also interesting are the highlights on the bottles which show similar structure in Fig. 6.7 and Fig. 6.8, but are actually smoother in the rendered image, while the highlights in the background of the rendering show faint stripes of red and blue, possibly an aliasing artifact, not present in Fig. 6.8.

Figure 6.8: A non-synthetic large aperture capture, obtained with a regular camera. Parameters differ from the rendered image in Fig. 6.7, but the effects in out of focus areas are similar, which shows the high realism achieved with the rendering in Fig. 6.7.

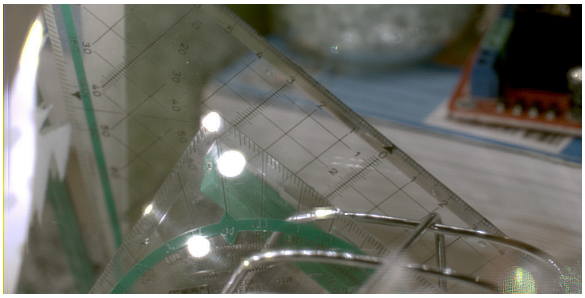
6.3.2 Viewpoint Sampling

Note that the targeted horizontal viewpoint sampling distance was slightly missed, with an average horizontal viewpoint spacing of 443 µm. Therefore all scene contents with a distance of less than 686 mm from the camera may be considered under sampled. This has to be kept in mind when evaluating the dataset and renderings produced from it. Although only the outer region falls under this, this includes most of the illumination. However, it should not be forgotten that the estimation of sufficient sampling is based on the MTF50 value, and not a strict boundary. As mentioned in Section 2.3.4 the definition of resolution using MTF50 is, while common, not the only possible measure for resolution. Indeed aliasing is present for the whole scene, but the magnitude of aliasing components are very small in the center, increasing with decreasing distance to the camera and reach a magnitude half that of the full signal at a distance of 686 mm from the center. However this increase in aliasing is a smooth process and

(a) Rendering produced from a single input image, using perspective reprojection.



(b) Clean image rendered from 5302 input images using the method from Section 5.7.4.2.



(c) The image from Fig. 6.9b sharpened using simple 2D sharpening, obtained using `GIMP` sharpen with a strength of 85.

Figure 6.9: Fig. 6.9a to Fig. 6.9c demonstrate the effectiveness of the enhanced resolution rendering. While a single image obtains high noise and low resolution, see Fig. 6.9b, the weighted projection scheme serves to produce a clean and high resolution result, see Fig. 6.9b. While this results in a very clean image, resolution is obviously limited by the resolution of the optical system. However, regular 2D image sharpening can be used to produce a subjectively higher resolution image, thanks to the low noise level. Note that this introduces artifacts, mainly in the defocused foreground and background, due to the assumption of constant depth.

not a sudden transition. No aliasing artifacts due to viewpoint undersampling have yet been discovered in renderings, though the undersampled area is also quite small and with simple content, not prone to show aliasing.

6.3.3 Enhanced Resolution Rendering

Fig. 6.9 highlights the effect of the enhanced resolution rendering, see Section 5.7.4.2, by comparing it with a single image from the dataset. Just like the regular synthetic aperture rendering the method assumes a constant depth, which in this mode is used to obtain sub-pixel accurate weighting at the presumed depth. But where the regular synthetic aperture rendering

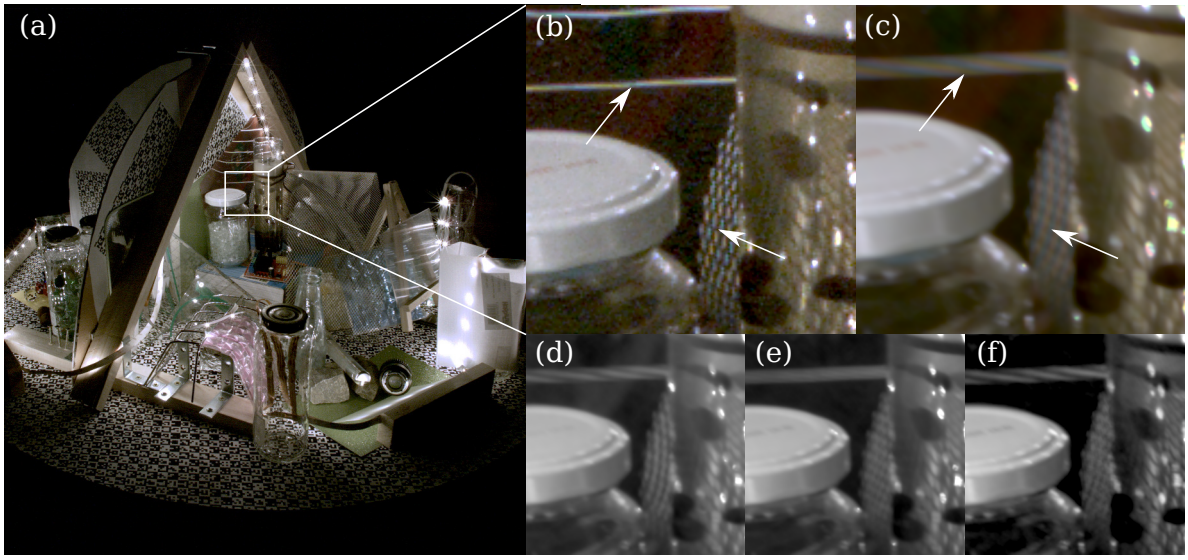
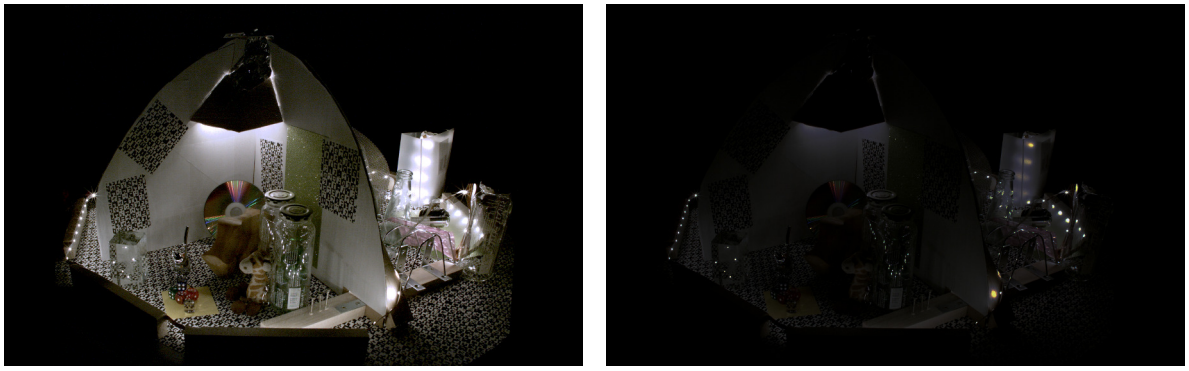


Figure 6.10: Several renderings from a single viewpoint: (a) shows the full scene, rendered from a single input image, (b) shows a crop from (a) highlighting aliasing from a lattice, visible as regular stripes of blue and red, see arrows. The aliasing is even visible in out of focus areas of a synthetic aperture rendering (c). (d)-(f) show the red, green and blue color channels of the the synthetic aperture rendering. The green channel (e) is sampled sufficiently and does not display aliasing while red (d) and blue (f) show strong aliasing. The effect is also visible in the reflection above the lattice.

produces smooth out of focus areas, the subpixel accurate weighting leads to a random weighting of unrelated samples, as the weight depends on the position on the reference plane. These artifacts are clearly visible in Fig. 6.9c. However at the correct depth this method exposes the full resolution of the light field dataset in the form of a very clean rendering, providing the basis for a subjectively higher resolution image, using simple 2D image sharpening. Note that the rendered image was sharpened in an 8 bit format, a higher bit depth could reduce sharpening artifacts somewhat.

6.3.4 Bayer Pattern Sampling

As mentioned in Section 5.1.5, for a Bayer sensor the sampling rate of the individual colors has to be taken into account. Indeed, for red and blue, the resolution of the optical system exceeds that of the sensor resolution which provides the sampling, see Section 5.1.5. This means that the red and blue channel may exhibit aliasing while the green channel does not, an effect which is clearly visible in some of the rendered images, see Fig. 6.10.



(a) Normal exposure **(b)** Underexposed rendering

Figure 6.11: A demonstration of the high dynamic rang of the dataset. Note that areas which are overexposed in the left image can be resolved in the right, underexposed image. Reproduction of the full dynamic range would require tonemapping or compression of the full dynamic range.

6.3.5 Signal Quality

In Fig. 6.9a it is apparent that individual frames exhibit a high amount of noise. Thanks to the non-linear sensor response, see Section 5.4.1, the dynamic range of the capture dataset is quite high. But most of the signal is observed in the lower part of the camera response, up to a relative radiance of around 0.05, relative to Fig. 5.5, and therefore obtains a higher amount of noise. However for many light field tasks, which fuse a high number of viewpoints to produce the desired output, noise is not a large problem, while the high dynamic range capture looses less of the original signal in saturated samples. Fig. 6.11 shows two samples from the dataset, rendered with different simulated exposures, to show the high dynamic range of the dataset.

7 Discussion and Future Work

This chapter presents a discussion of the relevance of the findings made in this work, as well as the connection to related fields. Also introduced are possible future works based on the findings of this thesis. The chapter starts with a short summary of the work in Section 7.1. Section 7.2 highlights the different viewpoints from which light field sampling is often regarded in the literature, while Section 7.3 discusses applications of dense sampling and of the dense light field dataset. Section 7.4 provides thoughts on the resolution tradeoff between different domains when capturing a light field. Section 7.5 regards fast compression under the aspect of memory transfer speeds, while Section 7.6 discusses fiducial marker detection.

7.1 Summary

The primary objective of this work was the examination and implementation of dense light field sampling. For sampling of bandunlimited 4D light fields, the bandlimiting effects of the optical system were examined in Section 2.3.6, showing that appropriate sampling rates for alias free sampling can be obtained from the bandlimiting properties of the optical system. A large light field dataset was recorded according to these calculations, demonstrating the feasibility of dense sampling for such a dataset. The possibility to render high quality synthetic aperture renderings from such a dense dataset, without geometric reconstruction or viewpoint interpolation, was demonstrated in Section 6.3.

Additionally, a fast lossless image compression method was presented, which provides more than an order of magnitude faster compression than the fastest dedicated image or video compression method to date, while obtaining a higher compression ratio than the fast generic compression methods, see Section 6.1.

Regarding fiducial marker detection, a system was designed and implemented, which provides a higher density of reference points than previous approaches, while providing a low overhead error detection through a well defined layout, see Section 5.6. At the same time high accuracy refinement for the reference points is incorporated, without decreasing the density of packing. The system also allows a much higher number of markers to be addressed and identified, increasing the number of usable reference points.

7.2 Perspectives on Light Field Sampling

Previous works have approached the problem of light field sampling from two different angles, either more theoretical or more pragmatic. The theoretical approach concentrates on the light field itself, examining the light field signal, with the actual capture process mostly simplified to a pinhole camera, and using synthetic scenes for demonstration and evaluation. This approach is representative of the works which actually derive minimal sampling rates, based on the light field signal itself [CTCS00, ZC03, DMMV05]. Aside from very limited scene contents, those works lead to the conclusion that light fields are in general bandunlimited, which makes light field sampling problematic.

On the other hand there are the more practical approaches, which do not concern themselves much with the theoretical sampling rates required for alias free sampling, but concentrate on the practical effects, removing or concealing artifacts due to undersampling by various post-processing methods. These works include all the practical demonstration of light field rendering and processing, as well as light field related methods [LH96, GGSC96, LPC+00, UWH+03, SYGM03, LCV+04, VGT+05, WJV+05, BZF09, GZC+06, KZP+13]. Note that this is not a comprehensive list, but includes all the concerned works cited throughout this thesis.

Finally, the compressive sensing approaches [VRA+07, LLW+08, AN10, BAL+12, MWBR13] provide a potential handle for dense sampling, avoiding undersampling while still keeping the number of samples low, on the basis of a sound theoretical framework [Don06]. However the actually achieved absolute resolutions seem still quite low, when compared to a multi-camera approach [MWBR13]. Also estimations for the theoretical resolution and sampling limits are not provided.

In this light, this work should be regarded under the aspect of bridging the gap, between the implications of a bandunlimited signal from the theoretical considerations on one side, and practical light field recording on the other side. This work examines the connection between these two, in the form of the optical system. This optical system mediates the actual sampling process, and, by virtue of its filtering properties on the light field signal, provides a practical low pass filter, which actually allows correct sampling of the bandunlimited light field signals predicted by the theoretical approaches, with the finite number of samples necessary for a practical implementation.

7.3 Applications to Dense Sampling

Light field sampling is normally regarded under the provision of undersampling. Indeed, compressive sensing, see Section 2.2.3, gives a framework which allows reconstruction from undersampled captures, exploiting the high redundancy in light fields, while viewpoint interpolation allows high quality renderings from few viewpoints. While those approaches are

necessary for more practical light field imaging, they hinge on the correct modeling of the light field contents, which can become problematic for complex scenes, as recorded in this work. The dense light field data set captured in this work may allow the evaluation of undersampling and compressive sensing, using the captured dataset as reference. Also the evaluation of processing and rendering strategies can be performed against ground truth renderings, produced from the densely sampled dataset.

Dense sampling at this scale requires a large amount of data, as well as long recording times, making it impractical, in the foreseeable future, for other applications than reference work and evaluation. A possible application outside this realm could be the capture of light fields for small static objects, with cultural or historical significance, allowing complete visual inspection without direct access to the subject.

7.4 On Resolution and Sampling

The sampling rates for the dense sampling of light fields are derived from properties of the optical system, by regarding the optical resolution and the aperture, which are the key parts to control the required sampling rates. Therefore a more throughout consideration of these two aspects of the optical system may give insights into the properties of light fields and sampling, and may pave the way for a range of interesting applications.

For example, as long as the sensor resolution is high enough to sample the full bandlimited 2D signal reaching the image sensor, arbitrary reduction of the resolution of the signal is possible in post processing with the appropriate image space filter. As the sampling rate required for alias free rendering depends on the distance from the camera this could be used to obtain high resolution samples for features far away from the camera and low resolution, but alias free samples for nearer features. Of course this requires the estimation of image depth, which can be difficult for complex scenes, but may make sense within an adaptive scheme. Basically this is a variant of compressive sensing, but approaches the problem from another angle, reconstructing low resolution alias free parts of the signal, and enhancing the signal by higher frequency content if it can be determined that aliasing does not exist.

When regarding the angular resolution, correct sampling of a circular aperture requires a viewpoint spacing that is smaller than the aperture. This has implications for example on the design of a camera array. Even completely dense packing cannot, in theory, eliminate aliasing, therefore some other technique is required, possibly using compressive sensing, or alternatively using semitransparent mirrors to allow the placement of viewpoints within the aperture of others. However, the angular component may not be very critical. Some works argue that the angular resolution is in general much lower than the spatial resolution [GZC+06], though detailed studies on different contents and on the perception of angular resolution would be useful.

Considering both domains of light field sampling, there is a fundamental, albeit indirect, connection between the angular and spatial trade off, for regular optics, by connection of the aperture. Small apertures induce diffraction blur, limiting spatial resolution, while large apertures reduce angular resolution. For a practical recording the depth of field also limits the possible trade off, although this could be avoided by using, for example, focus stacking.

7.5 Fast Lossless Image Compression

Due to the simple nature of the BBP compression scheme, the method reaches a very high speed, in the order of the absolute limit, set by the speed of memory transfer. While the integer compression schemes in [LB12] slightly surpass the speed of memory transfer, BBP reaches only half this speed. The differences can be attributed to the more complex processing required for image compression.

Regarding speed, as the method approaches the speed of a memory transfer, this conversely means that memory transfer attributes to a larger part of the processing time. However, the size of memory transfers is one of the parts which are actually influenced by the compression. A higher compression ratio means less memory to transfer. Therefore, for applications limited by the bandwidth of the processor, like for example image or light field processing, it would be interesting to investigate the effects of a tight integration between compression and processing. If data is processed in chunks small enough to fit in the L1 cache then compression could be used on the fly, reducing the bottleneck of memory transfer. Of course the trade off for such optimizations varies, and access patterns are not always easy to break down into predictable chunks, but for some workloads this might improve performance in situation where memory or storage capacity are not the limiting factors.

While BBP expands the envelope of compression methods towards higher speed, it cannot deliver the same compression ratio as other methods. THe method only exploits 2D correlation within image an image, plus a very simple form of 3D correlation for videos. The 4D nature of light fields offers one more dimension which could be used to improve compression. Also while the 3D method demonstrates that a 3D extension is possible, it is very suboptimal in implementation, reaching a relatively low speed at an only slightly improved compression ratio. A possible fast solution, using disparity compensation for better 3D correlation, as well as for 4D correlation, could be based on the PatchMatch algorithm [BSFG09]. The PatchMatch algorithm provides fast approximate per pixel correspondence mapping, by using random match candidates and propagation of good matches from neighboring pixels. In a video or light field coding system, the encoder could try an even smaller number of candidates and propagate good matches in the corresponding dimensions. As a dense light field dataset provides only relatively slow changes over several frames, the PatchMatch method could converge, while requiring few processing resources. Also with a synced pseudo random number generator

between encoder and decoder, the only additional information that needs to be transmitted is which of the few candidates was chosen for disparity compensation, reducing the overhead of such a method.

7.6 High Density Fiducial Markers

The `HDMarker` fiducial marker system introduced in this work has a range of advantages over previous systems, providing more reference points at a higher density, by incorporating a simple error detection which can recognize a large amount of errors. But this comes at the price of greatly reduced speed. However, when comparing the `HDMarker` system to other markers, the main difference is the method of error detection, a process which actually takes a negligible amount of processing time. This means that an alternative system which keeps the error detection methods, but incorporates the detection process of a faster method, should be able to achieve comparable speeds. This would reduce the accuracy and robustness of this approach somewhat, but might still be able to provide better detection than previous methods due to the reduced size and therefore increased detecteability.

Regarding applications of the `HDMarker` system, at the possible density of up to one marker per 25 pixels in the optimal case, it becomes feasible to improve distortion correction by using a more accurate model for the distortion, exploiting the high amount of reference points. In theory, the number of images necessary to get multiple reference points per pixel lies in the range of around a hundred images, and would provide per pixel model-less 2D distortion correction.

On a related note square markers with several reference points per marker also allow the estimation of distance from a marker to the camera, if marker size is known, which could additionally allow the modeling of depth dependent lens distortion.

The implemented marker detection allows markers to become arbitrarily large, as long as they still fit into the image, and allows smaller markers to be detected than previously possible. But if the camera position cannot be controlled, markers can be either too small for detection, or they may be very large and provide only few reference points. A possible scheme to remove those constraints could be provided by including markers at multiple scales. The scheme proposed in this work uses square pixels inside the marker for identification. If individual pixels are not simply black or white squares, but executed as individual markers then the range of detectable marker sizes could be improved. However such a recursive or fractal marker scheme would require a change of the marker layout and of the detection scheme, as saddle points, which are used in the current scheme for detection and refinement, could not be guaranteed to exist for the smaller scales.

8 Conclusion

As shown in this thesis, practical dense light field sampling is possible, even for large light fields, and allows artifact free rendering without viewpoint interpolation and on complex light fields. As light fields are in general not bandlimited the practical sampling is based on the derivation of the resolution of the light field, filtered by the respective optical system required for recording. The dataset produced in this work may prove useful for evaluation and as a reference for light field processing task and sampling methods.

The amount of data which needs to be capture for dense capture is quite high, and even the 22 TiB dataset recorded in this work cannot cover the complete hemisphere of the test scene. On the other hand the implemented `BBP` compression, capable of the compression of image, video and light field data at a rate which is over an order of magnitude faster than previous methods, allowed the storage of the recorded dataset on the available 12 TiB storage array and reduced the high I/O load for capture and processing, enabling faster capture and processing.

Additionally the dense marker scheme provided as part of this work, provides the required accurate calibration and features a very high number of precisely detectable reference points.

It may be hoped that the recorded dataset can foster research in dense, high resolution light field imaging and provide valuable raw data for the evaluation of light field processing and sampling tasks.

A Appendix

A.1 Documentation of Tools

Program usage and options, also available as help message from the respective program. For an overview of the functionality and tools, see Section 5.7.

A.1.1 lfbrowser

```
[light field tools - lfbrowser]
Provides visual examination of raw light field data
in lff format, in the form of a video player.
May also be used to extract frame ranges in the form
of images, see -w option.
usage:
lfbrowser [options] -l <input>
options:
-w     : don't display anything, write images to files
of the form img#########.png using the frame number
-d     : demosaic frames, default is display of the raw (Bayer) image
-p <int> : pause in ms between displaying the next frame
-s <int> : process only every <int> frames
-b <int> : begin processing at frame <int>
-e <int> : end processing at frame <int> start back at the beginning
-h     : print this help message
Playback loops over the full file or the selected range.
In playback mode [space] key pauses and unpauses playback,
while right and left arrow key change the direction of playback.
```

A.1.2 lfextract

```
[light field tools - lfextract]
Extracts frames and the respective camera position from provided .lff and .lfi file.
Also calculates average consecutive viewpoint distance.
usage:
lfextract [options] -l <file> -i <file>
options:
-l <file> : input light field file in .lff format
-i <file> : file containing the extrinsic calibrations in .lfi format
-p <file> : plot file, save the respective camera positions
-s <int> : process only every <int> frames
-b <int> : begin processing at frame <int>
```

```
-e <int> : end processing at frame <int>
-r <image> : substract dark frame
-h      : print this help message
If a plot file is given with -p then camera position is saved in a
gnuplot compatible format, and images are saved to imgs/#########.png.
If no plot file is provided than also no image is extracted, but the
distance between consecutive viewpoints is calculate and the average returned.
```

A.1.3 lfrecord

```
[light field tools - lfrecord]
Fast light field recording tool, only compatible with m3api from XIMEA
basic usage:
lfrecord <file>
Records to the specified file until stopped with ctrl-c
If <file> has the extension .tiff then only one frame is
recorded to the file and the program exists.
```

A.1.4 lfavg

```
[light field tools - lfavg]
synthethic aperture rendering and interactive light field rendering.
usage:
lfavg [options] -l <light field> -i <extrinsics> -c <intrinsics>
options:
-l <file> : input light field file in .lff format
-i <file> : file containing the extrinsic calibrations in .lfi format
-c <file> : intrinsic camera calibration, produced with lfcalibrate
-o <file> : batch mode, program exits after image is rendered and stored in <file>
-q <int> : number of image to integrate before updating the view
-x <float> : focal plane forward/backward rotation
-y <float> : focal plane right/left rotation
-d <float> : focal plane distance fromt origin
-a <float> : aperture size in radians
-z <float> : zoom
-b <file> : darkframe
-b <file> : darkframe
-r <float> : maximum weight for inverse distance weighting
-e <float> : exposure compensation (multiplier)
-s <int> : interpolation: 0 - demosaicing+bilinera, 1 - shepard, 2 - nearest neighbor (fast)
-p <int> : output size
-n <float> : use only a fraction of the frames (undersampling)
-k <float> : batch mode, renders a video, turning <float> degrees per frame, while a pseudo
         EPI image is created from the video. The number of frames depends on frame size,
-m <float> <float> : offset within the viewport
-v <float> <float> <float> : start with camera in this direction, lookin inwards
-t <float> <float> <float> : batch mode, render video and pseudo EPI, moving from -v to -t
-v <float> <float> <float> : start with camera in this direction, lookin inwards
-w <float> <float> <float> : white balance (multiplier)
-h      : print this help message
description:
If -o is not specificed the application starts in an interactive mode, rendering a viewpoint
```

```
somewhere in the middle of the dataset. In this mode parameters may be changed with the
keyboard. Any change restarts rendering with the new parameters. While rendering the view
is refreshed at least every -q frames, more often at the beginning or if IO is too slow.
Per default the focal plane stays at the same position relative to the camera, just like
a normal camera. with the 'r' key it can be toggled to stay at the same absolute position.
The following shortcuts are available. Most parameters may be changed in two steps, a
capital letter triggers the smaller step:
wWaAsSdD - rotate focus plane around scene center
arrow keys - rotate camera around scene center
qQeE    - move focal plane towards/away from the camera
[]      - change render refresh rate
-+      - change aperture
op      - zoom
ui      - change maximum weight for inverse distance weighting
kl      - toggle trough interpolation methods
r       - switch focal plane reference between scene and camera
```

A.1.5 lfinfo

```
light field tools - lfinfo]
reads extrinsic calibration data for a light field and
saves it into a text file.
usage:
lfinfo [options] -i <file> -p <file>
options:
-i <file> : file containing the extrinsic calibrations in .lfi format
-p <file> : output file, extrinsics will be saved in a gnuplot compatible format
-s <int> : process only every <int> frames
-b <int> : begin processing at frame <int>
-e <int> : end processing at frame <int>
-h      : print this help message
```

A.2 MTF Plots

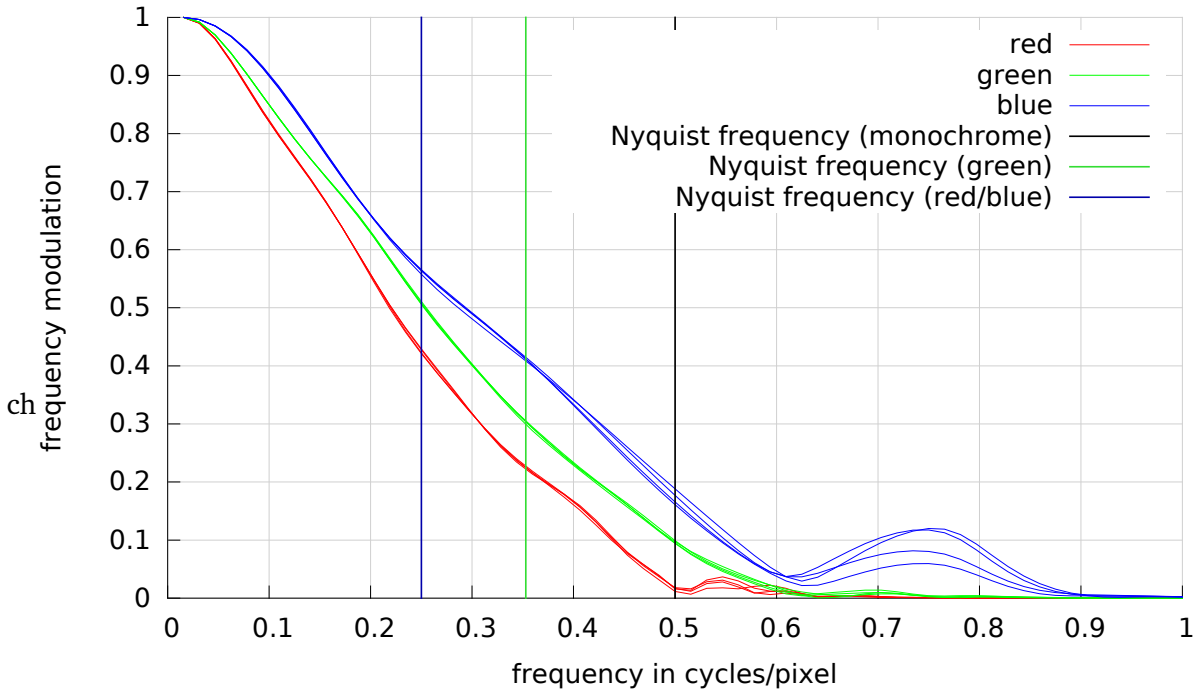


Figure A.1: This MTF plot shows the result of an optimal diffraction limited lens, simulated at an aperture of $f/16$, see Section 5.1.2.

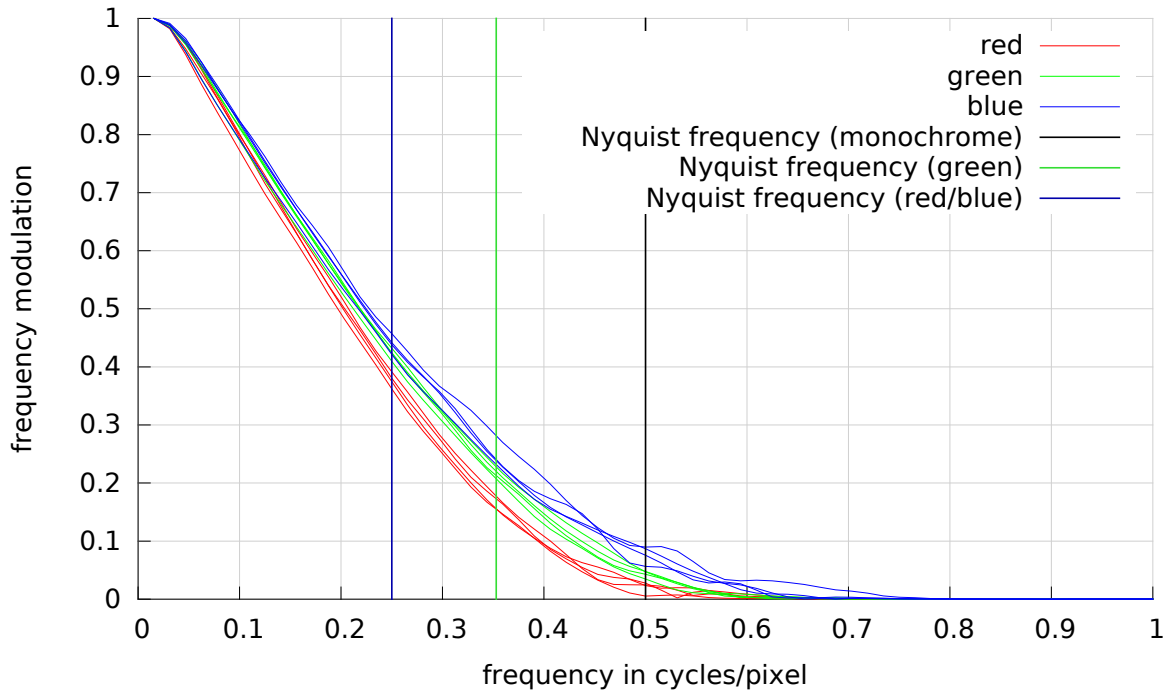
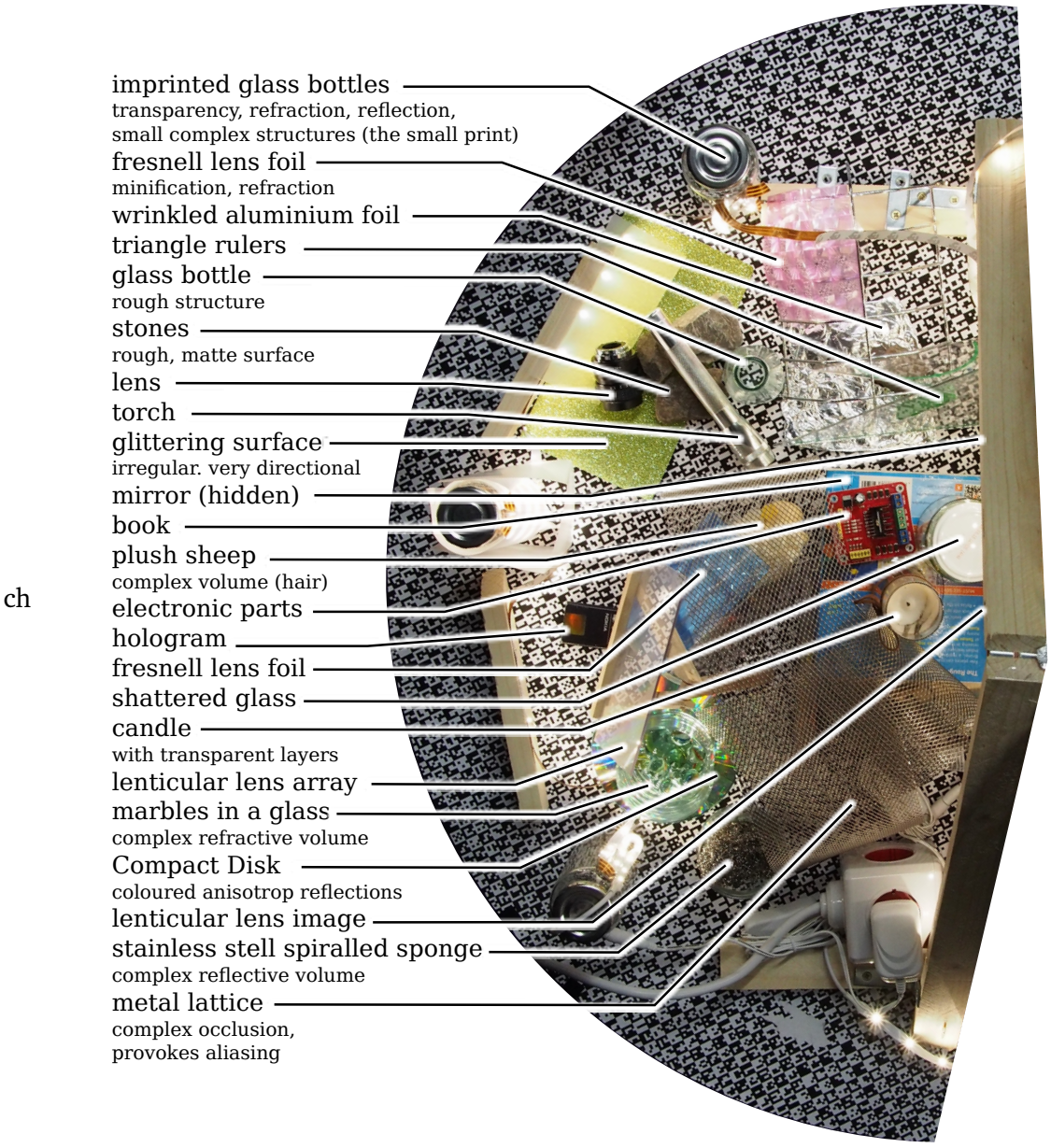


Figure A.2: A MTF plot of the lens used in this work, measured at $f/16$. Compared to the results at $f/8$ in Fig. 5.1b the effects of diffraction are now more pronounced, ordering the color channels by wavelength.

A.3 Contents of the Recorded Light Field



ch

Figure A.3: This image shows the large compartment from the recorded scene, viewed from above, highlighting the various scene elements and some optical properties.

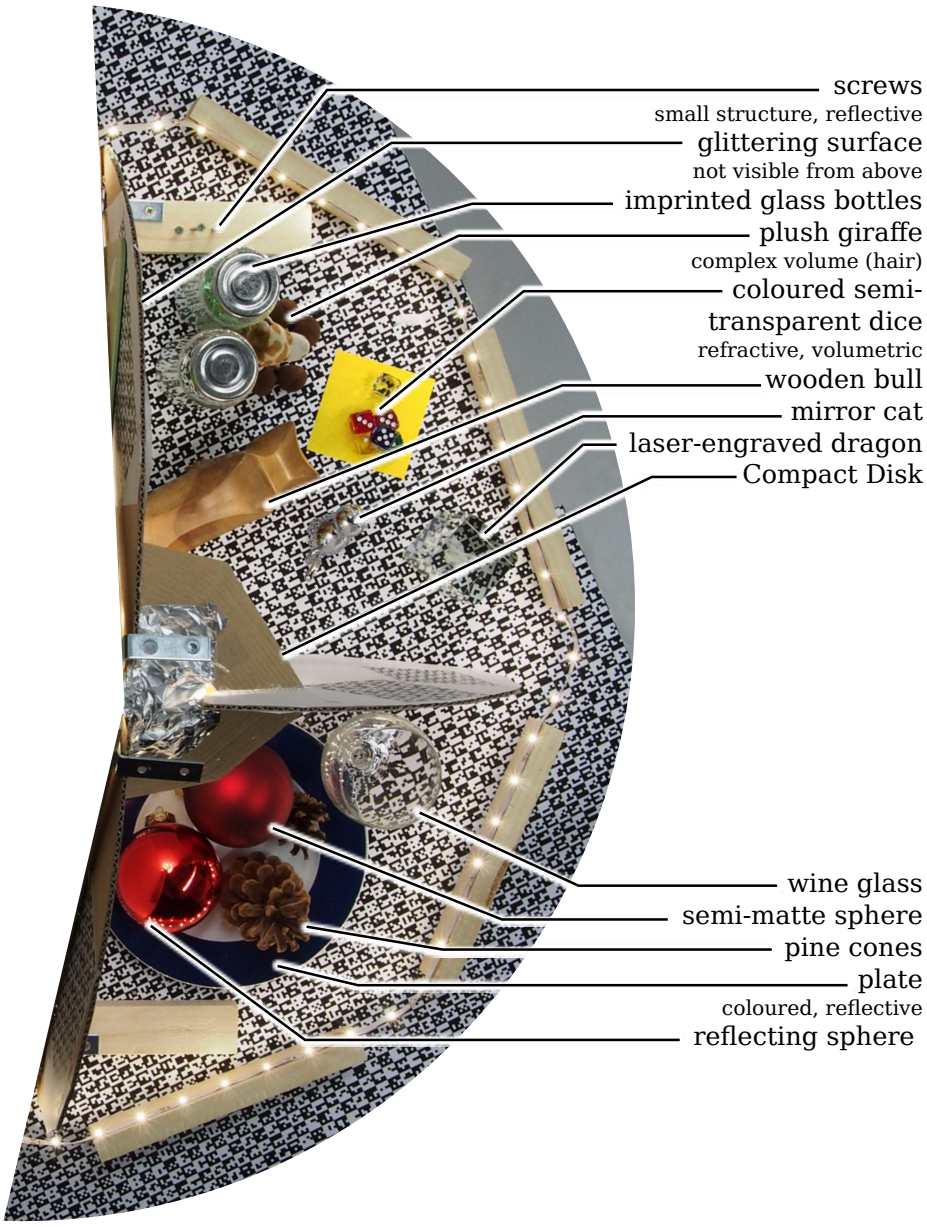


Figure A.4: This image shows the two small compartments from the recorded scene, viewed from above, highlighting the various scene elements and some optical properties.

Index

Airy pattern, 27
Aliasing, 31
ANS, 39
anti-aliasing, 32
aperture, 25
arithmetic coding, 38
Asymmetric numeral system, 39

back focal length, 25
bandlimited, 31
Bayer pattern, 33
BBP, 76
bitpacking, 76
block wise bitpacking, 76

C/P, 32
calibration matrix, 23
camera matrix, 23
chief ray, 25
chromatic aberration, 26
circle of confusion, 26
CoC, 26
color filter array, 33
color space, 39
color subsampling, 41
compressive sensing, 30
cutoff frequency, 32
cycles per pixel, 32

demosaicing, 33
depth of field, 26
diffraction, 27
DOF, 26

entropy coding, 37
extrinsic calibration, 24
extrinsic parameters, 23

fiducial markers, 48
focal length, 25
focal plane, 25
focal point, 25
Fourier transform, 31
FOV, 60
frequency multiplexing, 30
frequency substitution, 79
front focal length, 25

geometric distortion, 26
GOP, 102
group of picture, 102

HDR, 74
high dynamic range, 74
Homogeneous coordinates, 21
Huffman, 38

instantaneous code, 38
inter frame, 82
intrinsic parameters, 23

lateral chromatic aberration, 27
light field, 28
line pairs per millimeter, 32, 68
longitudinal chromatic aberration, 27
low pass filter, 32
Lp/mm, 32

magnification, 25

modulation transfer function, 32
MTF, 32

nodal point, 22
Nyquist frequency, 31
Nyquist rate, 31

perspective-3-point problem, 24
perspective-n-point, 24
plenoptic function, 28
PnP, 24
point spread function, 32
pose estimation, 24
prefiltering, 32
prefix code, 38
prefix free code, 38
PSF, 32

radiance, 28
RAID, 62
range coding, 38
Rayleigh criterion, 32
refractive index, 26
reprojection error, 24
RGB, 39
RMS, 24
root mean square, 24
rotation invariance, 49

sampling, 31
sampling theorem, 31
self identifying markers, 48
SIMD, 38
single instruction multiple data, 38
spatial multiplexing, 30
spectrum, 31

temporal multiplexing, 28
thin Lens, 25

universal codes, 38

variable length code, 38

YUV, 40

Bibliography

[AB91] E. H. Adelson, J. R. Bergen. The Plenoptic Function and the Elements of Early Vision. In *Computational Models of Visual Processing*, pp. 3–20. MIT Press, 1991. (Cited on pages 17 and 28)

[AGS11] L. Alvarez, L. Gómez, J. Sendra. Accurate Depth Dependent Lens Distortion Models: An Application to Planar View Scenarios. *Journal of Mathematical Imaging and Vision*, 39(1):75–85, 2011, doi: 10.1007/s10851-010-0226-2. (Cited on pages 26 and 107)

[AHH10] B. Atcheson, F. Heide, W. Heidrich. CALTag: High Precision Fiducial Markers for Camera Calibration. In R. Koch, A. Kolb, C. Rezk-Salama, editors, *Vision, Modeling, and Visualization (2010)*. The Eurographics Association, 2010, doi: 10.2312/PE/VMV/VMV10/041-048. (Cited on pages 50, 64 and 84)

[AN10] A. Ashok, M. A. Neifeld. Compressive light field imaging. In *Proceedings of the SPIE*, volume 7690, pp. 76900Q–76900Q–12. 2010, doi: 10.1117/12.852738. (Cited on pages 30 and 116)

[BAL+12] S. Babacan, R. Ansorge, M. Luessi, P. Mataran, R. Molina, A. Katsaggelos. Compressive Light Field Sensing. *Image Processing, IEEE Transactions on*, 21(12):4746–4757, 2012, doi: 10.1109/TIP.2012.2210237. (Cited on pages 30 and 116)

[Ber] F. van den Bergh. MTF Mapper. URL `http://sourceforge.net/projects/mtfmapper/`. (Cited on page 66)

[BL14] S. Bennett, J. Lasenby. ChESS - Quick and Robust Detection of Chess-board Features. *Comput. Vis. Image Underst.*, 118:197–210, 2014, doi: 10.1016/j.cviu.2013.10.008. (Cited on pages 89 and 90)

[BR09] M. Burtscher, P. Ratanaworabhan. FPC: A High-Speed Compressor for Double-Precision Floating-Point Data. *Computers, IEEE Transactions on*, 58(1):18–31, 2009, doi: 10.1109/TC.2008.131. (Cited on pages 47 and 63)

[BSFG09] C. Barnes, E. Shechtman, A. Finkelstein, D. B. Goldman. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Trans. Graph.*, 28(3):24:1–24:11, 2009, doi: 10.1145/1531326.1531330. (Cited on page 118)

[BZF09] T. Bishop, S. Zanetti, P. Favaro. Light field superresolution. In *Computational Photography (ICCP), 2009 IEEE International Conference on*, pp. 1–9. 2009, doi: 10.1109/ICCPHOT.2009.5559010. (Cited on pages 34 and 116)

[CAP+12] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, M. Cifrek. A brief introduction to OpenCV. In *MIPRO, 2012 Proceedings of the 35th International Convention*, pp. 1725–1730. 2012. URL http://docs.opencv.org/. (Cited on pages 22 and 94)

[CKB+14] C. Conti, P. Kovacs, T. Balogh, P. Nunes, L. Ducla Soares. Light-field video coding using geometry-based disparity compensation. In *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), 2014*, pp. 1–4. 2014, doi: 10.1109/3DTV.2014.6874724. (Cited on page 48)

[CNS12] C. Conti, P. Nunes, L. Soares. New HEVC prediction modes for 3D holoscopic video coding. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pp. 1325–1328. 2012, doi: 10.1109/ICIP.2012.6467112. (Cited on page 48)

[Col11] Y. Collet. LZ4 explained, 2011. URL http://fastcompression.blogspot.co.at/2011/05/lz4-explained.html. (Cited on page 47)

[CSE00] C. Christopoulos, A. Skodras, T. Ebrahimi. The JPEG2000 Still Image Coding System: An Overview. *IEEE Trans. on Consum. Electron.*, 46(4):1103–1127, 2000, doi: 10.1109/30.920468. (Cited on page 40)

[CTCS00] J.-X. Chai, X. Tong, S.-C. Chan, H.-Y. Shum. Plenoptic Sampling. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pp. 307–318. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000, doi: 10.1145/344779.344932. (Cited on pages 34 and 116)

[Deu96] P. Deutsch. DEFLATE Compressed Data Format Specification version 1.3. RFC 1951, 1996. URL https://tools.ietf.org/html/rfc1951. (Cited on page 45)

[DMMV05] M. Do, D. Marchand-Maillet, M. Vetterli. On the bandlimitedness of the plenoptic function. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pp. III–17–20. 2005, doi: 10.1109/ICIP.2005.1530317. (Cited on pages 32, 34 and 116)

[Don06] D. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, 2006, doi: 10.1109/TIT.2006.871582. (Cited on pages 30 and 116)

[Dud09] J. Duda. Asymmetric numeral systems. *CoRR*, abs/0902.0271, 2009. URL http://arxiv.org/abs/0902.0271. (Cited on page 39)

[FFM] A complete, cross-platform solution to record, convert and stream audio and video. URL https://www.ffmpeg.org. (Cited on pages 47 and 94)

[Fia10] M. Fiala. Designing Highly Reliable Fiducial Markers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1317–1324, 2010, doi: 10.1109/TPAMI.2009.146. (Cited on page 49)

[Fot] M. Fotheringham. The 135 STF. URL http://www.the135stf.net/. (Cited on page 37)

[FT03] fischertechnik GmbH, last retrevied 2015-01-03. URL http://www.fischertechnik.de/. (Cited on page 69)

[Fuj] Fujifilm Corporation. FUJINON LENS XF56mmF1.2 R APD. (Cited on page 37)

[GCAJ+13] R. D. Gomes, Y. G. G. d. Costa, L. L. Aquino Júnior, M. G. d. Silva Neto, A. N. Duarte, G. L. d. Souza Filho. A Solution for Transmitting and Displaying UHD 3D Raw Videos Using Lossless Compression. In *Proceedings of the 19th Brazilian Symposium on Multimedia and the Web*, WebMedia '13, pp. 173–176. ACM, New York, NY, USA, 2013, doi: 10.1145/2526188.2526228. (Cited on page 47)

[Ger36] A. Gershun. Light Field. In *Journal of Mathematics and Physics*, volume XVIII, pp. 51–151. MIT, 1936. (Cited on page 17)

[GGSC96] S. J. Gortler, R. Grzeszczuk, R. Szeliski, M. F. Cohen. The lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 43–54. ACM, New York, NY, USA, 1996, doi: http://doi.acm.org/10.1145/237170.237200. (Cited on pages 17, 28, 29, 48 and 116)

[GJSMCMJ14] S. Garrido-Jurado, R. Muñoz Salinas, F. J. Madrid-Cuevas, M. J. Marín-Jiménez. Automatic Generation and Detection of Highly Reliable Fiducial Markers Under Occlusion. *Pattern Recogn.*, 47(6):2280–2292, 2014, doi: 10.1016/j.patcog.2014.01.005. (Cited on pages 48, 49, 50 and 84)

[Goo96] J. Goodman. *Introduction to Fourier Optics*. MaGraw-Hill, 2nd edition, 1996. (Cited on page 32)

[GZC+06] T. Georgeiv, K. C. Zheng, B. Curless, D. Salesin, S. Nayar, C. Intwala. Spatio-angular Resolution Tradeoffs in Integral Photography. In *Proceedings of the 17th Eurographics Conference on Rendering Techniques*, EGSR '06, pp. 263–272. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2006, doi: 10.2312/EGWR/EGSR06/263-272. (Cited on pages 17, 30, 116 and 117)

[Hid11] A. Hidayat. FastLZ - lightning fast compression library, 2011. URL http://fastlz.org/. (Cited on page 47)

[JW76] F. Jenkins, H. White. *Fundamentals of Optics*. International student edition. McGraw-Hill, 1976. (Cited on page 26)

[KB99] H. Kato, M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on*, pp. 85–94. 1999, doi: 10.1109/IWAR.1999.803809. (Cited on pages 48 and 49)

[KGS98] V. A. Knyaz, H. O. Group, R. V. Sibiryakov. The Development Of New Coded Targets For Automated Point Identification And Non -. In *3D Surface Measurements, International Archives of Photogrammetry and Remote Sensing, Vol. XXXII, part 5*, pp. 80–85. 1998. (Cited on page 48)

[KZP+13] C. Kim, H. Zimmer, Y. Pritch, A. Sorkine-Hornung, M. Gross. Scene Reconstruction from High Spatio-angular Resolution Light Fields. *ACM Trans. Graph.*, 32(4):73:1–73:12, 2013, doi: 10.1145/2461912.2461926. URL http://www.disneyresearch.com/project/lightfields/. (Cited on pages 44 and 116)

[LA12] R. Lenhardt, J. Alakuijala. Gipfeli - High Speed Compression Algorithm. In *Proceedings of the 2012 Data Compression Conference*, DCC '12, pp. 109–118. IEEE Computer Society, Washington, DC, USA, 2012, doi: 10.1109/DCC.2012.19. (Cited on page 47)

[LB12] D. Lemire, L. Boytsov. Decoding billions of integers per second through vectorization. *CoRR*, abs/1209.2137, 2012. URL http://arxiv.org/abs/1209.2137. (Cited on pages 38, 47, 48, 63, 76, 78, 81 and 118)

[LCV+04] M. Levoy, B. Chen, V. Vaish, M. Horowitz, I. McDowall, M. T. Bolas. Synthetic aperture confocal imaging. *ACM Transactions on Graphics*, 23:825–834, 2004, doi: 10.1145/1015706.1015806. (Cited on pages 17 and 116)

[LH96] M. Levoy, P. Hanrahan. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 31–42. ACM Press, New York, NY, USA, 1996, doi: http://doi.acm.org/10.1145/237170.237199. (Cited on pages 17, 28, 29, 56 and 116)

[LLW+08] C.-K. Liang, T.-H. Lin, B.-Y. Wong, C. Liu, H. H. Chen. Programmable Aperture Photography: Multiplexed Light Field Acquisition. *ACM Trans. Graph.*, 27(3):55:1–55:10, 2008, doi: 10.1145/1360612.1360654. (Cited on pages 30 and 116)

[LNA+06] M. Levoy, R. Ng, A. Adams, M. Footer, M. Horowitz. Light field microscopy. *ACM Transactions on Graphics (Proc. SIGGRAPH 2006)*, 25(3):924–934, 2006, doi: http://doi.acm.org/10.1145/1141911.1141976. (Cited on page 43)

[LPC+00] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, D. Fulk. The Digital Michelangelo Project: 3D Scanning of Large Statues. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pp. 131–144. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 2000, doi: 10.1145/344779.344849. (Cited on pages 43, 56 and 116)

[LS99] M. Levoy, J. Shade. A light field of Michelangelo's statue of Night, 1999. URL http://graphics.stanford.edu/projects/mich/lightfield-of-night/. (Cited on pages 44 and 56)

[LSOJ14] Y. Li, M. Sjostrom, R. Olsson, U. Jennehag. Efficient intra prediction scheme for light field image compression. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 539–543. 2014, doi: 10.1109/ICASSP.2014.6853654. (Cited on page 48)

[Mac02] D. J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002. (Cited on page 38)

[MKMS07] R. Mantiuk, G. Krawczyk, R. Mantiuk, H.-P. Seidel. High Dynamic Range Imaging Pipeline: Perception-Motivated Representation of Visual Content. SPIE, pp. 649212–649212. SPIE, 2007, doi: 10.1117/12.713526. (Cited on page 75)

[MWBR13] K. Marwah, G. Wetzstein, Y. Bando, R. Raskar. Compressive Light Field Photography Using Overcomplete Dictionaries and Optimized Projections. *ACM Trans. Graph.*, 32(4):46:1–46:12, 2013, doi: 10.1145/2461912.2461914. (Cited on pages 30, 43 and 116)

[Nie13] M. Niedermayer. FFV1 Video Codec Specification, 2013. URL http://www1.mplayerhq.hu/~michael/ffv1.html. (Cited on page 47)

[NLMBH05] R. Ng, M. Levoy, M. H. Mathieu Bré, P. Hanrahan. Light Field Photography with a Hand-held Plenoptic Camera. *Stanford University Computer Scienece Tech Report*, 2005. (Cited on page 30)

[OAHY99] F. Okano, J. Arai, H. Hoshino, I. Yuyama. Three-dimensional video system based on integral photography. *Optical Engineering*, 38(6):1072–1077, 1999. (Cited on page 30)

[Obe] M. F. Oberhumer. oberhumer.com: LZO real-time data compression library. URL http://www.oberhumer.com/opensource/lzo/. (Cited on page 47)

[Ols11] E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3400–3407. 2011, doi: 10.1109/ICRA.2011.5979561. (Cited on pages 50 and 106)

[OPE] OpenJPEG library : an open source JPEG 2000 codec. URL http://www.openjpeg.org/. (Cited on page 101)

[OPE03] OpenBeam USA, last retrevied 2015-01-03. URL http://www.openbeamusa.com/. (Cited on page 69)

[Pav] I. Pavlov. LZMA SDK. URL http://www.7-zip.org/sdk.html. (Cited on page 45)

[PW13] J. Peatross, M. Ware. Physics of Light and Optics. Optical Society of America, 2013. URL http://optics.byu.edu/. (Cited on pages 27 and 32)

[RBP03] Raspberry Pi, last retrevied 2015-01-03. URL http://www.raspberrypi.org/. (Cited on pages 70 and 98)

[RC11] R. B. Rusu, S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, 2011. URL http://www.pointclouds.org/. (Cited on page 94)

[Rei11] L. M. Reinhold. Fast compression library for C, C# and Java, 2011. URL http://www.quicklz.com/. (Cited on page 47)

[RPI03] RPIO is an advanced GPIO module for the Raspberry Pi, last retrevied 2015-01-03. URL https://github.com/metachris/RPIO/. (Cited on page 98)

[RPN91] S. E. Reichenbach, S. K. Park, R. Narayanswamy. Characterizing digital image acquisition devices. *Optical Engineering*, 30(2):170–177, 1991, doi: 10.1117/12.55783. (Cited on page 32)

[Say05] K. Sayood. *Introduction to Data Compression, Third Edition (Morgan Kaufmann Series in Multimedia Information and Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. (Cited on pages 37, 38 and 39)

[SBGD07] J. Sattar, E. Bourque, P. Giguere, G. Dudek. Fourier tags: Smoothly degradable fiducial markers for use in human-robot interaction. In *Computer and Robot Vision, 2007. CRV '07. Fourth Canadian Conference on*, pp. 165–174. 2007, doi: 10.1109/CRV.2007.34. (Cited on page 48)

[Sew96] J. Seward. bzip2 and libbzip2, 1996. URL http://www.bzip.org. (Cited on page 45)

[Sha48] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, 1948, doi: 10.1002/j.1538-7305.1948.tb01338.x. (Cited on page 31)

[SHK+14] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. *German Conference on Pattern Recognition (GCPR 2014)*, pp. 31–42, 2014, doi: 10.1007/978-3-319-11752-2_3. (Cited on pages 17 and 44)

[SNA] snappy - A fast compressor/decompressor. URL https://code.google.com/p/snappy/. (Cited on page 47)

[SQU] Squash - Compression Abstraction Library. URL http://quixdb.github.io/squash/. (Cited on page 94)

[SYGM03] J. Stewart, J. Yu, S. J. Gortler, L. McMillan. A New Reconstruction Filter for Undersampled Light Fields. In P. H. Christensen, D. Cohen-Or, S. N. Spencer, editors, *Rendering Techniques*, volume 44 of *ACM International Conference Proceeding Series*, pp. 150–156. Eurographics Association, 2003. URL http://dblp.uni-trier.de/db/conf/rt/rt2003.html#StewartYGM03. (Cited on pages 17, 34 and 116)

[Sze10] R. Szeliski. *Computer vision algorithms and applications*. 2010. URL http://szeliski.org/Book/. (Cited on pages 21, 22, 24, 25, 26, 27, 31 and 33)

[TAV+10] Y. Taguchi, A. Agrawal, A. Veeraraghavan, S. Ramalingam, R. Raskar. Axial-cones: Modeling Spherical Catadioptric Cameras for Wide-angle Light Field Rendering. *ACM Trans. Graph.*, 29(6):172:1–172:8, 2010, doi: 10.1145/1882261.1866194. (Cited on page 30)

[Tog03] R. Togni. Description of the HuffYUV (HFYU) Codec, 2003. URL http://multimedia.cx/huffyuv.txt. (Cited on page 47)

[Tuc97] A. B. Tucker, Jr., editor. *The Computer Science and Engineering Handbook*. CRC Press, Inc., Boca Raton, FL, USA, 1997. (Cited on page 31)

[Uns00] M. Unser. Sampling-50 years after Shannon. *Proceedings of the IEEE*, 88(4):569–587, 2000, doi: 10.1109/5.843002. (Cited on pages 32 and 36)

[UWH+03] J. Unger, A. Wenger, T. Hawkins, A. Gardner, P. Debevec. Capturing and Rendering with Incident Light Fields. In *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pp. 141–149. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003. (Cited on pages 30 and 116)

[VGT+05] V. Vaish, G. Garg, E.-V. Talvala, E. Antunez, B. Wilburn, M. Horowitz, M. Levoy. Synthetic Aperture Focusing Using a Shear-Warp Factorization of the Viewing Transform. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops - Volume 03*, CVPR '05, pp. 129–. IEEE Computer Society, Washington, DC, USA, 2005, doi: 10.1109/CVPR.2005.537. (Cited on pages 17 and 116)

[VRA+07] A. Veeraraghavan, R. Raskar, A. Agrawal, A. Mohan, J. Tumblin. Dappled Photography: Mask Enhanced Cameras for Heterodyned Light Fields and Coded Aperture Refocusing. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07. ACM, New York, NY, USA, 2007, doi: 10.1145/1275808.1276463. (Cited on pages 30 and 116)

[Wal] P. van Walree. Distortion. URL http://toothwalker.org/optics/distortion.html. (Cited on page 26)

[Wan14] S. Wanner. *Orientation Analysis in 4D Light Fields*. Dissertation, IWR, Fakultät für Physik und Astronomie, Univ. Heidelberg, 2014. URL http://www.ub.uni-heidelberg.de/archiv/16439. (Cited on page 44)

[WB01] D. R. Williams, P. D. Burns. Diagnostics for Digital Capture Using MTF. In *Image Processing, Image Quality, Image Capture Systems Conference*, pp. 227–232. 2001. (Cited on page 32)

[WILH11] G. Wetzstein, I. Ihrke, D. Lanman, W. Heidrich. Computational Plenoptic Imaging. *Computer Graphics Forum*, 30(8):2397–2426, 2011, doi: 10.1111/j.1467-8659.2011.02073.x. (Cited on page 28)

[WJV+05] B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, M. Levoy. High performance imaging using large camera arrays. *ACM Trans. Graph.*, 24:765–776, 2005, doi: 10.1145/1073204.1073259. (Cited on pages 17, 29, 43 and 116)

[WKG+12] Z. Wang, M. Klaiber, Y. Gera, S. Simon, T. Richter. Fast lossless image compression with 2D Golomb parameter adaptation based on JPEG-LS. In *Proceedings of the 20th European Signal Processing Conference, EUSIPCO 2012, Bucharest, Romania, August 27-31, 2012*, pp. 1920–1924. 2012. (Cited on page 47)

[WMG13] S. Wanner, S. Meister, B. Goldluecke. Datasets and Benchmarks for Densely Sampled 4D Light Fields. In M. Bronstein, J. Favre, K. Hormann, editors, *VMV 2013: Vision, Modeling & Visualization (poster track)*, pp. 225–226. Eurographics Association, Lugano, Switzerland, 2013, doi: 10.2312/PE.VMV.VMV13.225-226. (Cited on pages 44 and 56)

[WS07] D. Wagner, D. Schmalstieg. ARToolKitPlus for Pose Tracking on Mobile Devices. In *Proc. 12th Computer Vision Winter Workshop (CVWW '07)*, pp. 139–146. 2007. (Cited on page 50)

[WSS00] M. Weinberger, G. Seroussi, G. Sapiro. The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS. *Image Processing, IEEE Transactions on*, 9(8):1309–1324, 2000, doi: 10.1109/83.855427. (Cited on page 47)

[X26] VideoLAN - x264, the best H.264/AVC encoder. URL http://www.videolan.org/developers/x264.html. (Cited on pages 47 and 101)

[X2617] x265 HEVC High Efficiency Video Coding H.265 Encoder, last retrevied 2014-12-17. URL http://x265.org/. (Cited on pages 47 and 101)

[xim03] ximea support. APIs - XiAPI, last retrevied 2015-01-03. URL http://www.ximea.com/support/wiki/apis/XiAPI. (Cited on page 95)

[ZC03] C. Zhang, T. Chen. Spectral analysis for sampling image-based rendering data. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(11):1038–1050, 2003, doi: 10.1109/TCSVT.2003.817350. (Cited on pages 32, 34 and 116)

[ZFS02] M. Zobel, M. Fritz, I. Scholz. Object Tracking and Pose Estimation Using Light-Field Object Models. In *Vision, Modeling, and Visualization Conference*. 2002. (Cited on page 56)

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000, doi: 10.1109/34.888718. (Cited on pages 24, 48 and 95)

[ZL77] J. Ziv, A. Lempel. A universal algorithm for sequential data compression. *Information Theory, IEEE Transactions on*, 23(3):337–343, 1977, doi: 10.1109/TIT.1977.1055714. (Cited on page 45)

[ZW06] N. Zhang, X. Wu. Lossless Compression of Color Mosaic Images. *Trans. Img. Proc.*, 15(6):1379–1388, 2006, doi: 10.1109/TIP.2005.871116. (Cited on pages 40, 41 and 101)

All links were last followed on January 10, 2015.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature