On Accuracy Analysis of 3D Scanners, Binarization, and Machine Learning

Research Thesis

In partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Ilya Blayvas

Submitted to the Senate of the Technion – Israel Institute of Technology

Tevet, 5771 Haifa

November, 2009
The research thesis was done under the supervision of Prof. Ron Kimmel, in the Faculty of Computer Science.

The generous financial help of Technion is gratefully acknowledged, as well as guidance and support by Prof. Ron Kimmel and Prof. Freddy Bruckstein.
## Contents

I Accuracy Analysis of Depth from Defocus and Depth from Structured Light 3D Scanners  13

1 Depth from Focus and Defocus Systems  14
   1.1 Overview  14
   1.2 Depth from focus and defocus  15
   1.3 Role of optics in the depth accuracy  17
      1.3.1 Spatial Domain analysis  17
      1.3.2 Fourier Domain analysis  23
      1.3.3 Depth from Defocus versus Depth from Focus  26
   1.4 The three dimensional light distribution near focus  27
   1.5 Revisiting depth reconstruction from defocus  30
   1.6 Chapter summary  34

2 Depth from Structured Light  36
   2.1 Overview  36
   2.2 Introduction  36
   2.3 The Image Sensor  40
      2.3.1 Electrical scheme and operation principles of CMOS pixels  41
2.3.2 Sensitivity and dynamic range . . . . . . . . . . . . . . 42
2.3.3 Noise and distortions . . . . . . . . . . . . . . . . . . . . 45
2.4 Image blur . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
2.5 Pattern projector . . . . . . . . . . . . . . . . . . . . . . . . 48
  2.5.1 Non-linearity and non-monotonicity . . . . . . . . . . . 48
  2.5.2 Pattern non-uniformity . . . . . . . . . . . . . . . . . . 49
  2.5.3 Temporal noise . . . . . . . . . . . . . . . . . . . . . . . 49
2.6 Ambient illumination, reflectivity and motion . . . . . . . . . 50
  2.6.1 Ambient illumination . . . . . . . . . . . . . . . . . . . 50
  2.6.2 Object reflectivity and motion . . . . . . . . . . . . . . 51
2.7 Revisiting depth from structured light . . . . . . . . . . . . . 52
2.8 Chapter summary . . . . . . . . . . . . . . . . . . . . . . . . 54

II Algorithms: Image Binarization, Machine Learning, and Dimensionality Reduction 56

3 Image Binarization 57
  3.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 57
  3.2 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 58
  3.3 Review of Binarization Methods . . . . . . . . . . . . . . . . 60
    3.3.1 Niblack’s Method . . . . . . . . . . . . . . . . . . . . . 60
    3.3.2 Eikvil-Taxt-Moen’s Method . . . . . . . . . . . . . . . 61
    3.3.3 Bernsen’s Method . . . . . . . . . . . . . . . . . . . . . 61
    3.3.4 Yanowitz-Bruckstein’s Method . . . . . . . . . . . . . . 61
    3.3.5 Analysis of the Binarization Methods . . . . . . . . . . 63
  3.4 The New Threshold Surface . . . . . . . . . . . . . . . . . . 66
    3.4.1 Interpolation Algorithm . . . . . . . . . . . . . . . . . 67
### 3.4.2 Approximating Source Function

3.5 Implementation

3.5.1 Data Structures

3.5.2 Algorithm description

3.6 Experimental results

3.7 Chapter summary

### 4 Machine Learning

4.1 Overview

4.2 Introduction

4.2.1 Machine Learning

4.2.2 Multiresolution Analysis

4.3 Multiresolution Approximation for Machine Learning

4.3.1 Prior Work

4.3.2 Our Approach

4.3.3 Analysis

4.4 Experimental Results

4.4.1 Pima Indians Dataset

4.4.2 Large Synthetic Dataset in 6D

4.4.3 Covertype Data

4.5 Chapter summary
List of Figures

1.1 The overall blur kernel, formed by convolution of the optical, defocus, and sampling kernels. ...................... 17
1.2 The defocus spot size. ..................................... 19
1.3 Step on the sharp pre-image, smoothed by the optics blur and defocus. .................................................. 20
1.4 Step-like profile, with different defocus blurrs. .......... 21
1.5 The difference between adjacent defocused images, as a function of defocus. ......................................... 22
1.6 Ratio between the distinguishable defocus spot $r_d$ and the optics blur $\lambda F$ as a function of signal to noise ratio. 23
1.7 Diffraction of a converging spherical wave near the focal point: Notation. ................................................. 27
1.8 Intensity profile in the vicinity of the focus. $u, v$ units. . . . . 29
2.1 Structured light 3D scanner. ................................ 39
2.2 Scheme of a single pixel in a typical CMOS Image Sensor. . . 42
3.1 Solution of the Laplace equation by the over-relaxation method 70
3.2 Interpolating surface, obtained by a new interpolation method. 70
3.3 The source function, given by (3.19) ....................... 73
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4</td>
<td>An approximating surface, obtained with source function (3.19)</td>
</tr>
<tr>
<td>3.5</td>
<td>Cell Hierarchy</td>
</tr>
<tr>
<td>3.6</td>
<td>Gray level image of 'Squares'</td>
</tr>
<tr>
<td>3.7</td>
<td>'Squares' binarized with YB method</td>
</tr>
<tr>
<td>3.8</td>
<td>'Squares' binarized with Multiresolution Approximation method</td>
</tr>
<tr>
<td>3.9</td>
<td>'Squares' binarized with Niblack method</td>
</tr>
<tr>
<td>3.10</td>
<td>Gray level image of 'Text'</td>
</tr>
<tr>
<td>3.11</td>
<td>'Text' binarized with YB method</td>
</tr>
<tr>
<td>3.12</td>
<td>'Text' binarized with Multiresolution Approximation method</td>
</tr>
<tr>
<td>3.13</td>
<td>'Text' binarized with Niblack method</td>
</tr>
<tr>
<td>3.14</td>
<td>Gray level image of 'Rectangles'</td>
</tr>
<tr>
<td>3.15</td>
<td>'Rectangles' binarized with YB method</td>
</tr>
<tr>
<td>3.16</td>
<td>'Rectangles' binarized with Multiresolution Approximation</td>
</tr>
<tr>
<td>3.17</td>
<td>'Rectangles' binarized with Niblack method</td>
</tr>
<tr>
<td>3.18</td>
<td>Gray level image of 'Stars'</td>
</tr>
<tr>
<td>3.19</td>
<td>'Stars' binarized with YB method</td>
</tr>
<tr>
<td>3.20</td>
<td>'Stars' binarized with Multiresolution Approximation method</td>
</tr>
<tr>
<td>3.21</td>
<td>'Stars' binarized with Niblack method</td>
</tr>
<tr>
<td>4.1</td>
<td>Multiresolution representation of a 2-dimensional feature space</td>
</tr>
<tr>
<td>4.2</td>
<td>Tree structure of the cells of a 2-dimensional feature space</td>
</tr>
</tbody>
</table>
List of Tables

1.1 Summary of reviewed depth from defocus publications (lengths in mm.). ......................................................... 34

3.1 Array pointarr. Column \( i \) contains the indices of the cells containing \( p_i \). .................................................................. 77

3.2 Array coeffs. Contains decomposition coefficients \( a_{ljk} \) and number of support points \( P_{ljk} \) in the cell \( ljk \). ............... 77

3.3 Comparison of the speeds and performance of the Yanowitz-Bruckstein (YB) and Multiresolution Approximation (MA) and Niblack binarization methods. .............................. 77

4.1 Experimental results for the Pima Indians dataset ......... 104

4.2 Experimental results for a 6D dataset, comparison of [1] and this work ................................................................. 106

4.3 Forest cover type. Classification matrix from [2], network 54-120-7, overall performance 70.58% ................................. 108

4.4 Forest cover type. Training, test times and classification performance and its standard deviation for different training set sizes and values of maximum resolution level ............. 110
4.5 Forest cover type. Classification matrix for Multiresolution Approximation. Training set of size 512 000, maximum resolution level 5. Overall performance 87.85 %.
Notation

Chapter 1

D  A lens diameter.
f  A focal length of the lens.
$F, F^\#$  F-number of the lens, $F \equiv f/D$.
h(x)  A point spread function.
$I(x)$  A focused image, formed from $s(x)$ after addition of the optics and sampling blurs and sensor noise.
$J(x)$  A real image, formed from $s(x)$ after addition of the optics, defocus and sampling blurs and sensor noise.
$J_1$  A Bessel function of the first order.
l  A distance from the lens to the image plane.
L  A distance from the object to the lens.
r_d  A radius of defocus spot.
$s(x)$  A sharp pre-image.
u, v  Spatial frequency parameters in the $x$ and $y$ directions.
$x = \{x, y\}$  A coordinate in the two dimensional image.
$\gamma$  $\gamma \equiv (\pi|x|D)/(\lambda l)$.
$\Delta L$  A minimal depth resolution.
$\Delta x$  A pixel size.
$\epsilon(x)$  An additive noise in the image sensor.
$\eta(x)$  A defocus kernel.
$\lambda$  A wavelength of the light.
$\mu(x)$  An optics blur kernel.
$\nu, \nu, \nu, \nu$  spatial frequencies of defocus, sampling, and optics blurs.
$\rho(x)$  A sampling kernel.
$\zeta$  $\zeta \equiv 2 \arccos\left(\lambda F \sqrt{u^2 + v^2}/2\right)$.
\( a = D/2 \)  A lens radius.

\( O \)  An axial focal point.

\( U(P) \)  Field intensity at arbitrary point \( P \).

\( \mathbf{R} \)  Vector from \( O \) to \( P \).

\( dS \)  Element of the wave-front.

\( d\Omega \)  Element of the solid angle.

\( W \)  Optical Front at the lens.

\( J_n \)  A Bessel function of order \( n \).
Chapter 2

$C$ A Camera.

$C_{\text{pix}}$ Electric capacitance of a pixel.

$d$ A stereo basis between pattern projector and camera.

$I_{\text{max}}$ A maximal pixel reading.

$j_{\text{dark}}$ Dark current, caused by the thermal leakage.

$L$ A distance to the object.

$n_e$ A number of electrons, generated in the pixel.

$n_{\text{ph}}$ A number of photons that reach the pixel.

$n_{\text{ph}}^{\text{min}}$ The number of photons, corresponding to the single bit of pixel reading.

$N$ Number of bits of a pixel output.

$N_e$ A maximum electron capacitance of the pixel.

$p_i$ Projected pattern number $i$.

$P$ A Pattern projector.

$T$ A temperature of the image sensor.

$V_{\text{max}}$ Voltage saturation of a pixel.

$\Delta L$ A minimal depth resolution.

$\Delta I$ A minimal step of the pixel reading.
$\eta_q$ Quantum efficiency of a pixel.
$\phi$ An angle to the camera axis.
$\Phi$ A camera angle of view.
$\psi$ An angle to the projector axis.
$\Psi$ A projector angle of view.
$\sigma_I$ An image intensity registration accuracy.
$\sigma_{n_e}$ A variance of the number of electrons.
$\sigma_T$ A temperature coefficient of the dark current.
$\sigma_x$ Overall blur scale on the image plane.
$\sigma_\phi$ An angular resolution of the camera.
$\sigma_\psi$ An angular resolution of the projector.

**Chapter 3**

$a_{ijk}$ A decomposition coefficient of the threshold surface.
$B(x,y)$ A binarized image.
$G_{ijk}(x,y)$ Basis function, shifted and down-scaled by one step.
$I(x,y)$ An original image.
$p_i = \{x_i, y_i\}$ Support point.
$T(x,y)$ A threshold surface.
$v_i = I(x_i, y_i)$ Value of the support point.
$x, y$ The image coordinates.
Chapter 4

$c^l_j$  Decomposition coefficient.

$C^l_j$  Cell of the feature space.

$D$  Dimensionality of the feature space.

$f_{\vec{X}, Y}(\vec{x}, y)$  A joint probability distribution function.

$j$  Spacial position index, $j \equiv \{j_1, \ldots, j_D\}$.

$l$  Resolution level.

$l_{\text{max}}$  Maximum resolution level.

$m$  $m = |T|$, Cardinality of a training set.

$M_\alpha(\vec{x})$  Classifier of a type $\alpha$.

$M(\vec{x})$  A classifier.

$R(M)$  Risk functional.

$T = \{\vec{x}_i, y_i\}_{i=1}^m$  Training set - a set of training points.

$T$  Tree of the decomposition coefficients.

$\{\vec{x}, y\}$  Training point.

$\{\vec{X}, Y\}$  Two dependent random variables.

$\vec{x}$  Training point coordinate (feature vector).

$y$  Value of a training point.
Abstract

This research report starts with accuracy analysis of optical 3D scanners. These scanners are becoming efficient and popular. Here, we focus on a specific type of such optical devices. A core algorithm in most of the scanners we analyze is a binarization procedure for which we dedicate the second part of this thesis. Finally, it appears that a generalization of the binarization procedure we propose can be used as an efficient classifier in the field of machine learning. We dedicate the last part of this document for that application.

We start from analysis of depth from focus/defocus and depth from structured light optical systems, which are the most common optical 3D scanners. A 3D scanning system accuracy is determined by the optical system, the physical properties of scanned object, and the processing algorithm used for reconstructing the geometry of the object. The analysis we provide demonstrates that, for all published results, the properties of the optical system have a crucial role in the final accuracy, while the processing algorithm is less significant. The accuracy estimators we derive allow us to predict an accuracy of a given system, or design a new one that would satisfy a set of given accuracy requirements.

Some publications describing new algorithms for 3D scanners and presenting experimental results do not provide the optical parameters. In these
cases, we obviously can not use our analysis of the whole system for a fair comparison between the algorithms. We therefore consider our contribution to be important at a pedagogical level. That is to say, in the field of computer vision, a reconstruction system should be evaluated only provided all optical parameters.

We start with analysis of optical systems for structured light 3D scanners, where we discuss both the role of optics as well as the image sensor in the accuracy of the whole system.

Structured light 3D scanners based on binary patterns are one of the popular 3D scanning techniques. Image binarization is an important step in these scanners, where the black and white regions of the projected pattern are extracted from the acquired gray level image. For that goal, we revisited the image binarization problem, and proposed our own analysis and improvement to the popular Yanovitz-Bruckstein binarization method.

Our binarization algorithm is based on efficient construction of an adaptive threshold surface for image binarization. The threshold surface is constructed as a multi-resolution interpolation of sparsely distributed support points. A generalization of the construction algorithm to a multidimensional spaces yields a multiresolution density estimator, that allows us to construct a computationally efficient Bayesian classifier, with constant time per training point and per query. The curse of dimensionality limits the application domain of this algorithm to $n \lesssim 10-15$, and therefore the problem of dimensionality reduction had naturally attracted our attention. For that goal, we analyze the limitations and applicability of linear dimensionality reduction via Generalized Singular Value Decomposition (GSVD).
Part I

Accuracy Analysis of Depth from Defocus and Depth from Structured Light 3D Scanners
Chapter 1

Depth from Focus and Defocus Systems

1.1 Overview

Depth from focus/defocus approach to 3D reconstruction is based on the fact that objects closer or further from the object in focus appear blurred, and the amount of blur increases with the distance from the object in focus. An important characteristic of any depth from defocus system is the depth reconstruction accuracy. Several 3D reconstruction algorithms have been proposed, and the influence of image noise and image spectrum on the system accuracy has been studied. However, so far the effect of optics on the accuracy has not been fully explored. Here, we re-derive an expression estimating the system accuracy as a function of its optical parameters, which is known as ‘depth of field’ in the optics [3, 4]. It turns out that optics plays a major role in the accuracy, and tenfold increase of the lens focal length and the aperture can increase the overall accuracy by a factor of more than one
thousand. The derived expression allows to review several published results, revealing that the accuracy is primarily defined by the optics. Our analysis also provides guidelines for the design of new depth from defocus systems in compliance with pre-defined specification by choosing the appropriate optics. This chapter is based on [5].

1.2 Depth from focus and defocus

Recovering 3D information from 2D images is an important task in computer vision. Among other approaches for 3D reconstruction, commonly referred to as “Depth from X”, where X stands for stereo, shading, motion, structured light etc. Depth From Defocus and closely related Depth From Focus approaches have attracted substantial attention [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. These approaches are based on the limited depth of view of the lens. When the lens is focused on a certain object, objects located closer or farther from the object in focus appear blurred. The blur increases with the distance from the object in focus. This phenomena allows to estimate the geometry of a scene by measuring the amount of blur in an image. In a depth from defocus approach, two images with different defocus levels are used. The depth from focus approach relies on multiple images, focused at different distances evenly distributed through the depth range. This gives the depth from focus approach some advantage in accuracy over depth from defocus at the expense of a larger number of the acquired images. The derivations we present here are relevant to both approaches.

An important characteristic of depth from defocus is the reconstruction accuracy, which depends on many factors, like camera settings [17], number of observations [18], image noise [19], and spatial frequencies [20]. Numerous
algorithmic ways to improve the reconstruction accuracy have been proposed [21, 22, 23, 24, 25, 26, 27].

Among the papers tackling the role of optics, most relevant to our discussion are [25] and [28]. Subbarao and Surya in [25] used a thin lens formula to derive a quadratic dependence of the reconstruction error on the distance $L$ to the object. However, they did not estimate the magnitude of the error. Nayar et al. in [28] decomposed an image blur into separate defocus, sensor, and optical transfer function blurs. However, the blurring model was used to design a linear operator for computing the relative blur of the two images, and not for the accuracy analysis.

Here, we follow [28] and decompose the blur into defocus, sensor, and optics blur. We extend the analysis by expressing the blur in terms of basic optical parameters and use the resulting model in order to estimate the system accuracy. The expression for the system accuracy confirms the quadratic dependence of the error on $L$, derived in [25].

The main contribution of this paper is an estimation of the system accuracy as a function of its optical parameters. We show that the system optics actually defines the reconstruction accuracy. For example, we show how the change of the lens focal length and the aperture by a factor of ten can result in a change of depth resolution by a factor of more than one thousand. Review of the published results from the perspective of this estimation confirms the fundamental role of the optics on the system accuracy.
1.3 Role of optics in the depth accuracy

1.3.1 Spatial Domain analysis

An image $J(x)$ captured by an image sensor is formed from a sharp pre-image $s(x)$, blurred by the optics, the defocus, and the finite size of the pixels. It also includes some noise. Formally, the image can be described via the convolution of $s(x)$ with the point spread function $h(x)$, as

$$J(x) = h(x) * s(x) + \epsilon(x),$$ \hspace{1cm} (1.1)

where, $x = \{x, y\}$ is a coordinate in the two dimensional image, and $\epsilon(x)$ describes an additive noise.

$h(x)$ can be considered as a convolution of the optical blur $\mu(x)$, the defocus blur $\eta(x)$, and the sampling blur $\rho(x)$ kernels [28], as shown on the figure 1.1,

$$h(x) = \int \int \rho(x-s)\eta(s-t)\mu(t)dsdt.$$ \hspace{1cm} (1.2)

Figure 1.1: The overall blur kernel, formed by convolution of the optical, defocus, and sampling kernels.
Accurate modelling of $\mu(x)$, $\eta(x)$, $\rho(x)$, is technically difficult and requires a special software and a detailed physical model of the lens and the sensor [29], which are usually unavailable.

The popular measure of lens quality is the Modulation Transfer Function, which is defined as the ratio of relative image contrast divided by relative object contrast, where the object is the sinusoidally varying brightness at some spatial frequency. The MTF depends on the location on the image plane, and decreases with increase of the spatial frequency. Despite the relation of MTF to the lens resolution, the blur kernel $\mu(x)$ can not be derived from it. Usually, as the $F$-number of the lens increases from low to high, the blur of the lens first decreases due to the improvement of the optical wavefront, and then increases due to increasing diffraction blur [30]. The blur for the lower end of the $F$-number range varies among different lenses, while the blur for the higher end of the $F$-number range of most decent lenses approaches the diffraction limit [30, 3],

$$
\mu(x) = \left[ \frac{2 J_1(\gamma)}{\gamma} \right]^2.
$$

Here, $\gamma = (\pi |x| D) / (\lambda l)$, $J_1$ is a Bessel function [31], $\lambda = 0.7 \cdot 10^{-6}$ m is the wavelength of the light, $l$ is the distance from the lens to the image plane, and $D$ is the lens diameter (see Fig. 1.2). In practical cases $l \approx f$, where $f$ is the focal length of the lens. Using the definition of the $F$-number, $F \equiv f / D$, we can write $\gamma \approx (\pi |x|) / (\lambda F)$.

The defocus kernel has a cylindrical shape given by

$$
\eta(x) = \begin{cases} 
\frac{1}{\pi r_d^2}, & |x| \leq r_d \\
0, & |x| > r_d 
\end{cases}
$$

(1.4)
Figure 1.2: The defocus spot size.

[4, 32], where the radius of the cylinder (see Fig. 1.2) is given by

\[
2r_d = \left| \frac{\Delta l D}{l^2} \right| = \left| \frac{\partial l}{\partial L} \frac{\Delta L D}{l^2 - 2} \right| = \left| \frac{f^2 \Delta LD}{L^2 - l^2} \right| \approx \left| \frac{f^2 \Delta L}{L^2 - f} \right|. \tag{1.5}
\]

Here, the derivative \( \partial l / \partial L \), is obtained from the thin lens formula [4], connecting between the focal length \( f \), the distance \( L \) to the object, and the distance \( l \) to the focused image of the object

\[
\frac{1}{l} + \frac{1}{L} = \frac{1}{f}. \tag{1.6}
\]

One should be careful in comparing the diffraction of the lens to the geometric optics analysis of the defocus [33]. In section 1.4 we will review the diffraction model for defocus based on [3], [34], and justify that geometric approximation is sufficient for our goals.

Finally, the sampling kernel describes averaging over a square pixel of size \( \Delta x \)

\[
\rho(x) = \begin{cases} 
\frac{1}{\Delta x^2} \max(\{|x_1|, |x_2|\}) \leq \frac{\Delta x}{2} \\
0 & \text{otherwise.}
\end{cases} \tag{1.7}
\]

Now consider an image in focus \( I(x) = \rho * \mu * s(x) \) and a defocused image \( J(x) = \rho * \eta * \mu * s(x) \). The minimal depth resolution \( \Delta L \) is the depth change,
at which the difference between the image in focus and the defocused image becomes distinguishable.

Fig. 1.3 presents results of computer simulation of the diffraction and defocus blurs. The sharp pre-image $s(x)$ has a step-like profile, shown as a dashed line. The focused image of this profile, blurred only by the optics diffraction with $\lambda F = 1$ is shown by a solid line $I = \mu(1) * s(x)$. The dotted lines show the defocused images, blurred by defocuses with $r_d = 1$ and $r_d = 2$; $J_1 = \eta(1) * \mu(1) * s(x)$ and $J_2 = \eta(2) * \mu(1) * s(x)$. Arbitrary small defocus blur will result in some difference between the focused and defocused blurs. In practical cases such difference is obscured by the image quantization and noise. The difference between the images should exceed the quantization and noise level, in order to be reliably detectable. The defocus might be
detectable also when the difference between the images is below the noise level. However, rigorous analysis of the minimal distinguishable difference between two noisy images seems to be a somewhat unrelated task. Actually the field of image denoising in image processing introduces many solutions, each one tailored for a specific scenario. Noise analysis requires assumptions about the nature of the noise and the image contents and is beyond the scope of this paper [17, 19] For the sake of simplicity, we assume that the noise exceeds the image quantization level, and that the minimal detectable difference between the images must be equal to the noise level. Furthermore, we assume that the sharp pre-image has features equivalent to a step with the magnitude 10% of the full intensity range. These assumptions may seem too pessimistic or the contrary, but even if they are false, all the further conclusions would still remain qualitatively same, although the numbers will
Figure 1.5: The difference between adjacent defocused images, as a function of defocus.

The maximum difference between the focused and defocused images, shown on the Fig. 1.5 are 0.0113 for an image blurred with $\eta(r_d = 1)$ and 0.0264, for an image blurred with $\eta(r_d = 1)$. They will become distinguishable from the focused image at the $S/N = 15.8dB$ and $S/N = 19.5dB$ respectively.

For any ratio $r_d/(\lambda F)$ there is a corresponding noise level at which the defocus becomes detectable. Fig. 1.6 shows $r_d/(\lambda F)$ as a function of the signal to noise of the image. The simulation was performed for an image with a step of 10% of the intensity range.
1.3.2 Fourier Domain analysis

In order to take into account the sampling blur (image pixelization) and derive an analytical estimate for the resolution as a function of optical parameters, we proceed to the analysis in the Fourier domain. In the Fourier domain any blur is a low-pass filter, that cuts off or significantly suppresses ‘high’ frequencies, starting from some characteristic frequency $\nu \approx 1/\zeta$, where $\zeta$ is the scale of the blur kernel in the spatial domain. In the previous section we have seen that at the reasonable signal to noise ratios, the scales of defocus and optical blurs must be comparable. This implies that in the Fourier domain their characteristic frequencies must be comparable.

In the Fourier domain, the diffraction blur is transformed into

$$
\mathcal{F}(\mu(x)) = \begin{cases} 
F^2 (\zeta - \sin \gamma), & \sqrt{u^2 + v^2} \leq \frac{2}{\lambda F} \\
0, & \sqrt{u^2 + v^2} > \frac{2}{\lambda F}
\end{cases},
$$

(1.8)
where $u$ and $v$ denote spatial frequency parameters in the $x$ and $y$ directions, respectively, and $\zeta = 2 \arccos \left( \frac{\lambda F \sqrt{u^2 + v^2}}{2} \right)$ [28]. The defocus blur then reads

$$F(\eta(x)) = \frac{L^2 F}{2 \pi f^2 \Delta L \sqrt{u^2 + v^2}} J_1 \left( \frac{2 \pi f^2 \Delta L}{L^2 F \sqrt{u^2 + v^2}} \right) ,$$

(1.9) [3, 28]. Finally, the sampling kernel blur is given by

$$F(\rho(x)) = \frac{1}{2 \pi} \frac{1}{\Delta v^2} \sin \frac{u \Delta x}{2} \sin \frac{v \Delta x}{2} .$$

(1.10)

Now, consider two images $I$ and $J$. Let us assume that the first image is in focus and the second one is out of focus. Then, $I(x) = h(x, L) * s(x)$ and $J(x) = h(x, L + \Delta L) * s(x)$. In the Fourier domain, we have,

$$F(I) = F(\rho(x))F(\mu(x))F(s(x))$$

$$F(J) = F(\rho(x))F(\eta(x))F(\mu(x))F(s(x)).$$

(1.11)

The minimal depth resolution $\Delta L_{\text{min}}$ is the value, at which the difference between $F(I)$ and $F(J)$ is detectable. The images $I$ and $J$ are blurred, and therefore they have a band-limited spectrum with a characteristic frequencies $\nu_I$ and $\nu_J$. We can say that $\nu_I \approx \min\{\nu_\rho, \nu_\mu, \nu_s\}$, $\nu_J \approx \min\{\nu_\eta, \nu_\rho, \nu_\mu, \nu_s\}$, where $\nu_{\eta, \rho, \mu, s}$ denotes respectively the characteristic frequencies of the band limited kernels $\eta, \rho, \mu$, and $s$. For the image in focus $I$, the defocus kernel $\eta$, given by (1.4) approaches a delta function, and does not limit the bandwidth of $I$, which is defined by diffraction and sampling blurs. As the image $J$ gets out of focus, the defocus radius $r_d$ increases, and the defocus blur first becomes comparable and then even exceeds the diffraction and sampling blurs. At this moment the second image becomes more blurred than the first one. Therefore, the images $I$ and $J$ become distinguishable when the defocus blur exceeds the diffraction and sampling blurs. In the frequency domain this means that the characteristic frequency of band-limited defocus kernel

28
becomes lower than characteristic frequencies of diffraction and defocus blurs: 
\[ \nu_\eta \approx \min\{\nu_\rho, \nu_\mu, \nu_s\} . \]

Here, we assume that the resolution is optical-limited and not limited by 
the absence of high spatial frequencies in the image: \( \nu_s > \{\nu_\rho, \nu_\mu\} \). In this 
case the defocus blur becomes distinguishable when

\[ \nu_\eta \approx \min\{\nu_\rho, \nu_\mu\} , \quad (1.12) \]

that can be rewritten as

\[
\frac{1}{\nu_\eta} \approx \max \left\{ \frac{1}{\nu_\rho}, \frac{1}{\nu_\mu} \right\} \Rightarrow \frac{1}{\nu_\eta} \approx \sqrt{\frac{1}{\nu_\rho^2} + \frac{1}{\nu_\mu^2}} . \quad (1.13)
\]

From the equations (1.8), (1.9), and (1.10), describing the spectrums of 
the kernels, we can estimate their characteristic frequencies as \( \nu_\eta = L^2 F / (2\pi f^2 \Delta L_{\text{min}}) \), 
\( \nu_\rho = 2 / (\Delta x) \), and \( \nu_\mu = 2 / (\lambda F) \). Substituting into (1.13), we obtain

\[
\frac{2\pi f^2 \Delta L_{\text{min}}}{L^2 F} \approx \sqrt{\left(\frac{\Delta x}{2}\right)^2 + \left(\frac{\lambda F}{2}\right)^2} ,
\quad (1.14)
\]

thereby,

\[
\Delta L_{\text{min}} \approx \frac{L^2 F}{2\pi f^2} \sqrt{\left(\frac{\Delta x}{2}\right)^2 + \left(\frac{\lambda F}{2}\right)^2} .
\quad (1.15)
\]

Again, we would like to stress that \( \Delta L_{\text{min}} \) measures the depth shift at which 
the defocus blur becomes comparable to the diffraction and sampling blurs. 
\( \Delta L_{\text{min}} \) can only serve as an estimate for the resolution of the system. In the 
case of low image noise, feature rich image, and efficient processing algorithm 
the resolution can, in principle, exceed \( \Delta L_{\text{min}} \). In the case of noisy and 
textureless image the resolution can be lower than \( \Delta L_{\text{min}} \).

Substituting some ‘typical’ numbers into (1.15), like \( \{f = 20\text{mm}, F = 16, L = 1\text{m}, \Delta x = 5\mu m\} \), we obtain \( \Delta L_{\text{min}} = 39\text{mm} \). Replacing the lens
for one with ten times longer focus and opening the aperture by a factor of
ten, like \{f = 200\text{mm}, F = 1.6, L = 1\text{m}, \Delta x = 5\mu m\}, we obtain \(\Delta L_{\text{min}} = 0.016\text{mm}\). Thus, we have shown that change of the lens can improve the
accuracy by a factor of \(~2500\).

In the literature, the popular figure of merit of an algorithm is the relative
error \(\frac{\Delta L_{\text{min}}}{L}\). We can see from (1.15) that

\[
\frac{\Delta L_{\text{min}}}{L} = \frac{LF}{2\pi f^2} \sqrt{\left(\frac{\Delta x}{2}\right)^2 + \left(\frac{\lambda F}{2}\right)^2}, \tag{1.16}
\]
depends on \(L, f, F\), and \(\Delta x\) - the optical parameters of the system, and
therefore the measure \(\Delta L_{\text{min}}/L\), often used in publications that explore the
depth from defocus problem, does not provide an optics-invariant comparison
between the algorithms.

### 1.3.3 Depth from Defocus versus Depth from Focus

The above analysis refers to the case where one image is focused, and the
second one is defocused, which is a depth from focus approach. In the depth
from defocus both images are defocused. In that case the achievable accuracy
degrades. Fig. 1.4 shows a step profile, blurred by different defocuses, with
\(r_d \in [0.2, 20]\). Fig. 1.5 shows the maximum differences between adjacent
profiles, \(\max(\eta(r_d + \Delta r_d)\mu(1)s(x) - \eta(r_d)\mu(1)s(x))/\Delta r_d\) One can see that
after a prompt increase, near \(r_d \approx 0\), the difference decays from its maximum
value of 0.02, near \(r_d \approx 2\) to \(~0.005\) and below near \(r_d \sim 20\) and beyond.
This figure illustrates, that the resolution of depth from defocus approach
gradually decades with increasing defocus, but remains proportional to the
resolution of the depth from focus approach.
1.4 The three dimensional light distribution near focus

One should be careful in comparing the diffraction of the lens to the geometric optics analysis of the defocus [33]. In this section we review the diffraction model for defocus based on [3], [34]. The numerical evaluation of the intensity field near the focus will justify that geometric approximation is sufficient for our goals.

Consider a spherical monochromatic wave emerging from a circular aperture and converging towards the axial focal point $O$, as shown on Figure 1.7. We shall consider the field intensity $U(P)$ at an arbitrary point $P$ in the neighborhood of $O$. The point $P$ will be specified by vector $\mathbf{R}$ relative to $O$.

![Figure 1.7: Diffraction of a converging spherical wave near the focal point: Notation.](image)

Let $A/f$ be the uniform amplitude of the incident wave. Then, by appli-
cation the Huygens-Fresnel principle, we obtain
\[
U(P) = -\frac{i}{\lambda} \frac{A e^{-ikf}}{f} \int \int_{W} e^{ikd} \frac{dS}{d}. \tag{1.17}
\]
If \( q \) is a unit vector in the direction \( OQ \), we have \( d - f = -qR \). Also the element \( dS \) of the wave-front is given by \( dS = f^2 d\Omega \), where \( d\Omega \) denotes the element of the solid angle that \( dS \) subtends at \( O \). Moreover, we can replace \( d \) by \( f \) in the denominator of the integrand, obtaining the Debye integral \[35\]
\[
U(P) = -\frac{i}{\lambda} A \int \int_{\Omega} e^{-ikqR} d\Omega. \tag{1.18}
\]
Introducing the dimensionless variables
\[
u = 2 \frac{a}{f} \zeta, \quad \mu = 2 \frac{a}{f} \sqrt{x^2 + y^2}, \tag{1.19}
\]
we have that \( \frac{\zeta}{\mu} \leq 1 \) defines if the point \( P \) lies in the geometric beam of light or in the geometric shadow. Then, after some algebraic manipulations \[3\], one obtains
\[
U(P) = -\frac{2\pi a^2 A}{\mu f^2} e^{i(\frac{\mu}{\nu})^2} \int_{0}^{1} J_0(v\xi) e^{-\frac{1}{2} iux^2} \xi d\xi = C(u, v) - iS(u, v), \tag{1.20}
\]
where \( J_0 \) is a Bessel function of order 0 \[31\], and
\[
\begin{align*}
C(u, v) &= \cos \frac{v}{\nu} U_1(u, v) + \sin \frac{v}{\nu} U_2(u, v) \\
S(u, v) &= \sin \frac{v}{\nu} U_1(u, v) - \cos \frac{v}{\nu} U_2(u, v)
\end{align*}
\tag{1.21}
\]
and
\[
\begin{align*}
U_n(u, v) &= \sum_{k=0}^{\infty} (-1)^k \left( \frac{u}{v} \right)^{n+2k} J_{n+2k}(v) \\
V_n(u, v) &= \sum_{k=0}^{\infty} (-1)^k \left( \frac{v}{u} \right)^{n+2k} J_{n+2k}(v)
\end{align*}
\tag{1.22}
\]
are the Lommel functions \[36\]. The intensity \( I = |U|^2 \) is given by
\[
I(u, v) = \left( \frac{2}{u} \right)^2 \left[ U_1^2(u, v) + U_2^2(u, v) \right] I_0, \tag{1.23}
\]
32
Figure 1.8: Intensity profile in the vicinity of the focus. $u, v$ units.
Figure 1.8 shows the numerical estimation of the intensity near the focus in the \((u,v)\) plane. The red lines show the geometric shadow of the light. The diameter of the white spot in the focus is \(1.22\lambda F = 1.22\lambda \frac{f}{2a}\), where \(\lambda\) is the wavelength, \(F = \frac{f}{2a}\) is the \(F\)-number of the lens, \(f\) is the focal length and \(2a\) is the lens aperture. One can see, that the defocus follows the profile of the geometric shadow even in the vicinity of the focal point, and it doubles the scale of the blur kernel at distance \(u \approx 1/2\). This observation justifies the use of geometric model of the defocus kernel, as proposed in (1.4,1.5).

1.5 Revisiting depth reconstruction from defocus

Most of the depth from defocus publications evaluate their accuracy as an RMS error divided by the distance range. As one can see from (1.16), this measure depends on the optics. We would like to compare between reported results in an optics-invariant way, by dividing each reported accuracy by its corresponding \(\Delta L_{\text{min}}\), derived from the reported optical parameters of the system. The ratio between the reported accuracy \(\Delta L_{\text{rep}}\) and its optical ‘limit’ \(\Delta L_{\text{min}}\) will be referred to as the accuracy factor, \(\alpha \equiv \Delta L_{\text{rep}}/\Delta L_{\text{min}}\). Estimation of \(\Delta L_{\text{min}}\) requires knowledge of \(L, f, F, \Delta x\). Unfortunately, some of these parameters are sometimes missing from the reports.

Among the relevant publications that we could not evaluate here due to the missing optical parameters are [28] with microscopic depth from defocus, where \(L, f, \Delta x\) are missing; [37] where \(L, \Delta L, \Delta x\) are missing; [25] and [38], where \(L, f, \Delta x\) are missing; and [23], where \(\Delta x\) is missing. Actually the pixel size \(\Delta x\) is not reported in most of the papers, but in some cases it
can be recovered, from the documentation of the camera.

Subbarao and Choi in [24] proposed to use the focused image surface in order to reconstruct the 3D shape of an object. They derived and experimentally confirmed that $\Delta L \sim L^2$, in compliance with (1.15). The optical parameters were $f = 35\text{mm}$, $F = 4$, $L = 1\text{m}$, and $\Delta x = 13\mu\text{m}$, while $\Delta L_{\text{rep}} = 3\text{cm}$. Substituting the optical parameters into (1.15), we obtain $\Delta L_{\text{min}} = 3.5\text{mm}$. Therefore, for this algorithm the accuracy factor is $\alpha = 8.6$.

Subbarao and Surya in [25] apply a spatial domain transform to extract the depth information from two images, each with different camera parameters. In their system $f = 35\text{mm}$, $F = 4$, $L = 2\text{m}$, $\Delta x = 13\mu\text{m}$, and $\Delta L_{\text{rep}} = 0.1\text{m}$. Substituting these figures into (1.15), we obtain $\Delta L_{\text{min}} = 13.8\text{mm}$, thereby $\alpha = 7.24$.

Ens and Lawrence in [22] calculate the depth information from two defocused images, acquired with two different $F$-numbers. Defocusing by change of the $F$-number preserves $l$ and allows to exclude an unwanted scaling of the image. The blur is treated as a convolution of a sharp image $s(x, y)$ with a low-pass filter $h(x, y)$. A ‘less blurred’ image $J_1(x, y) = s(x, y) * h_1(x, y)$ is acquired with a defocused system with higher $F$-number $F_1$, and a ‘more blurred’ image $J_2(x, y) = s(x, y) * h_2(x, y)$ with $F_2 < F_1$. Then, the inverse problem is solved to find a blurring function $h_3(x, y)$ which transforms $J_1(x, y)$ into $J_2(x, y)$, $J_2(x, y) = J_1(x, y) * h_3(x, y)$. In the Fourier domain this deconvolution problem translates into a simple division

$$F(h_3(x, y)) = \frac{F(J_2(x, y))}{F(J_1(x, y))}.$$

(1.24)

The one-to-one relation between $h_3(x, y)$ and the depth is derived from the geometric optics or found from a look-up table, evaluated on a calibration object.
A Javelin JE2062 CCD camera was used with focal length of \( f = 50 \) mm. The pixel size \( \Delta x \) was not reported. We could not recover the sensor type used in this camera, but from the camera focal length, field of view, and number of pixels, the pixel size can be estimated to be \( \Delta x \approx 50 \mu m \). The two different \( F \)-numbers were \( F_1 = 1.3 \) and \( F_2 = 2.0 \). The distance to the object was in the range of 80 to 95 cm, where we use an average of \( L = 87.5 \) cm.

Substituting these figures into (1.15), we obtain \( \Delta L_{\text{min}} = 1.8 \) mm.

The authors present several solutions to the inverse filtering problem (1.24), obtaining RMS error of 6.8\% for consistent inverse filtering and 1.7\% for iterative matrix solution, 1.3\% using experimentally measured blurring operators. For the average distance \( L = 87.5 \) cm this translates into \( \Delta L_{\text{rep}} = 60 \) mm, 15 mm, 11 mm respectively. Thus, for the best case, the accuracy factor is \( \alpha = 11 \text{mm}/1.8 \text{mm} = 6.1 \).

Watanabe and Nayar in [26] used a near- and far-focused images to extract the depth information. Let us denote by \( J_{\text{near}} \) and \( J_{\text{far}} \) the images with a non-zero frequency in the spatial domain. Then, the expression \( (J_{\text{near}} - J_{\text{far}})/(J_{\text{near}} + J_{\text{far}}) \), is a monotonic function of distance in between the focus range of near-focused image \([L_{\text{near}}, \text{and the focus range of the far-focused image}, L_{\text{far}}]\). In order to increase the processing speed, a small set of broadband filters derived by precisely modelling an image blur was used.

Sony XC-77 camera was used, with Cosmicar B1214D-2 lens. The optical parameters were \( f = 25 \text{mm}, L = 70 \text{cm}, \Delta x = 11 \mu m, \) and \( F = 8.3 \). For this optical setup \( \Delta L_{\text{min}} = 9.2 \) mm, while \( \Delta L_{\text{rep}} = 5 \) mm. Thus, the algorithm accuracy factor was \( \alpha = 5/9.2 = 0.54 \), which means that the the algorithm detects the defocus blur when it is still below the diffraction and sampling blurs. Such a good accuracy can be explained by the fact that for each of the two focus settings 256 images were averaged to reduce noise. Averaging over
256 images before the linear processing of depth reconstruction is equivalent to reconstructing first and then averaging of 256 depth maps, which increases the reconstruction accuracy.

Xiong and Shafer in [27] used a depth from focus approach, when the distance to the point is estimated by focusing on it. They used a flat black/white step edge as an artificial target to measure the depth resolution. Combination of Fibonacci search and curve fitting was used to detect the focus peak. The focal length was \( f = 130\text{mm} \), the distance to the object \( L \approx 1.2\text{m} \), the \( F \)-number \( F = 1.7 \), and the pixel size \( \Delta x = 23\mu\text{m} \). \( \Delta L_{\text{rep}} = 1.18\text{mm} \), while the accuracy estimated by (1.15) for this system is \( \Delta L_{\text{min}} = 0.22\text{mm} \). Therefore, \( \alpha = 1.18/0.22 = 5.36 \).

Baba et al. [21] estimated the depth accounting for zoom, focus, \( F \)-number, and the lens center transition. The authors used SONY XC-007 camera with the pixel size \( \Delta x = 50\mu\text{m} \) and a Canon J16x9.5B4RAS lens. The focal lengths were \( f = 9.5 - 152\text{mm} \), used at \( 40 - 130\text{mm} \) range, the \( F = 1.8 \), \( L \in \{1,1.5,2,2.5,3\}\text{m} \). The depth accuracy was \( \Delta L_{\text{rep}} \in \{22,45,71,93,112\}\text{mm} \) respectively, while the corresponding accuracies predicted by (1.15) for this system, with \( F = 70\text{mm} \), are \( \Delta L_{\text{min}} \in \{1.5,3.3,5.8,9.1,13.2\}\text{mm} \). The corresponding accuracy factors are \( \alpha_i = \{15,13.7,12.1,10.2,8.5\} \), respectively.

Table 1.1 summarizes the results reported in the reviewed papers, with focal length \( f \), distance to the object \( L \), \( F \)-number \( F \), sensor pixel size \( \Delta x \), and the reported accuracy of depth estimation \( \Delta L_{\text{rep}} \), all lengths in millimeters. The two right columns are the corresponding accuracy bound \( \Delta L_{\text{min}} \) and the algorithm accuracy factor \( \alpha = (\Delta L_{\text{rep}})/(\Delta L_{\text{min}}) \).
1.6 Chapter summary

We have shown that the optical parameters of a depth from defocus system - the focal length, the distance to the object, the \( F \)-number, and the sensor pixel size have a crucial role on the accuracy. A change of the focal length and the \( F \)-number by a factor of ten each, can increase the system depth resolution by a factor of more than a thousand.

Thus, comparing the published methods by reported accuracy, does not allow to separate between the role of optics and the algorithm. For example, the absolute resolution of five out of the six published results, reviewed in this paper, varied by a factor of 95 from 1.18\,mm to 112\,mm, their relative accuracies \( \Delta L_{\text{rep}}/L \) varied by a factor of 37 from 0.1\% to 3.7\%, while the resolution relative to the corresponding accuracy estimate varied only by a factor of 1.5. Therefore, normalization of the system resolution by the accuracy estimate \( \Delta L_{\text{min}} \), given in (1.15), reveals a relatively modest role of the reconstruction algorithm on the system performance. Equation (1.15) estimating the system accuracy as a function of optical parameters can help

<table>
<thead>
<tr>
<th>Ref.</th>
<th>( f )</th>
<th>( L )</th>
<th>( F )</th>
<th>( \Delta x )</th>
<th>( \Delta L_{\text{rep}} )</th>
<th>( \Delta L_{\text{min}} )</th>
<th>( \alpha )</th>
</tr>
</thead>
<tbody>
<tr>
<td>[24]</td>
<td>35</td>
<td>1000</td>
<td>4</td>
<td>0.013</td>
<td>30</td>
<td>3.5</td>
<td>8.6</td>
</tr>
<tr>
<td>[25]</td>
<td>35</td>
<td>2000</td>
<td>4</td>
<td>0.013</td>
<td>100</td>
<td>13.8</td>
<td>7.24</td>
</tr>
<tr>
<td>[22]</td>
<td>50</td>
<td>875</td>
<td>1.3</td>
<td>0.050</td>
<td>11</td>
<td>1.8</td>
<td>6.1</td>
</tr>
<tr>
<td>[26]</td>
<td>25</td>
<td>700</td>
<td>8.3</td>
<td>0.011</td>
<td>5</td>
<td>9.2</td>
<td>0.54</td>
</tr>
<tr>
<td>[27]</td>
<td>130</td>
<td>1200</td>
<td>1.7</td>
<td>0.023</td>
<td>1.18</td>
<td>0.22</td>
<td>5.36</td>
</tr>
<tr>
<td>[21]</td>
<td>70</td>
<td>3000</td>
<td>1.8</td>
<td>0.050</td>
<td>112</td>
<td>13.2</td>
<td>8.5</td>
</tr>
</tbody>
</table>

Table 1.1: Summary of reviewed depth from defocus publications (lengths in mm.).
to design depth from defocus systems in compliance with a given accuracy specification.
Chapter 2

Depth from Structured Light

2.1 Overview

The accuracy of a structured light 3D scanner depends on the characteristics of the camera, pattern projector, object reflectivity, motion, and ambient illumination. In this chapter we provide an accuracy estimation of 3D scanner systems as a function of these factors. This estimation enables quantification of the accuracy bounds for existing structured light 3D scanners and provides guidelines for the design of new systems.

2.2 Introduction

A structured light 3D scanner is similar to a pair of passive stereo vision cameras, where one of the cameras is replaced by a pattern projector [39]. An image pixel represents an ambiguity cone in 3D, while a row in the projected pattern is a plane in 3D. The intersection of a narrow cone (ray), and a plane allows us to reconstruct the coordinates of an interception point in 3D space.
Structured light 3D scanners are one of the most efficient optical acquisition devices [40]. Recently, they have attracted significant attention, and reviews of the field can be found in [41], [42], [43].

An important characteristic of a 3D scanner is its accuracy. Most of the published papers on 3D scanners discuss the accuracy of the proposed system. With the growing number of new publications that focus on mainly improvement of the registration process, there is an obvious need for a general, system-independent analysis.

In this chapter we present an accuracy analysis of a generic structured light 3D scanner. The accuracy is derived in terms of the characteristics of the camera, the projector, the ambient illumination, the distance to the object, its reflectivity and motion.

In [40], the range accuracy of a triangulation systems is considered, taking into account the diffraction of the optics. We will show, that the diffraction of the optics is indeed one of the important factors, that determine the system accuracy.

An early methodology for the design of an optimal pattern is presented in [32]. Here we do not elaborate on how to design such patterns, but rather provide a system accuracy estimation, where the pattern is given as one of the factors. In [44], the relation between the depth accuracy $\Delta L$ and the intensity registration accuracy $\sigma_I$ is explored. It actually corresponds to the second term of (2.6).

Various 3D scanning systems with different types of projected patterns have been introduced over the years [45], [46], [47], [22], [48], [49], [50], [51], [52], [53], [54], [55], [56], and [57]. Here we focus our discussion on the systems with gray level patterns. We have chosen gray level patterns because they are
an efficient and popular solution, capable of providing a dense depth map, with sub-millimeter reconstruction accuracy at a video rate [57]. However, the analysis provided here can be extended to systems with arbitrary patterns. All that is required is to estimate the angular resolution of the pattern $\sigma_\psi$, and to substitute the estimate into (2.5).

Figure 2.1 presents a generic structured light 3D scanner. $d$ denotes the stereo basis between the pattern projector $P$ and the camera $C$. $L$ is the distance to the object; $\psi$ and $\varphi$ are the angles from the axes of the projector and the camera to an object point.

We readily have that

$$L \tan (\varphi) = L \tan (\psi) + d,$$

thereby,

$$L = \frac{d}{\tan (\varphi) - \tan (\psi)},$$

(2.2)

Differentiating the distance with respect to the angles, we obtain

$$\frac{\partial L}{\partial \psi} = \frac{d}{(\tan (\varphi) - \tan (\psi))^2 \cos^2 (\psi)} \frac{1}{d \cos^2 (\psi)} = \frac{L^2}{d \cos^2 (\psi)};$$

(2.3)

and

$$\frac{\partial L}{\partial \varphi} = -\frac{L^2}{d \cos^2 (\varphi)}.$$  

(2.4)

Assuming independent variances of $\varphi$ and $\psi$, we can describe the variance of $L$ as

$$\Delta L = \frac{L^2}{d} \sqrt{\frac{\sigma^2_\varphi}{\cos^4 (\varphi)} + \frac{\sigma^2_\psi}{\cos^4 (\psi)}}.$$  

(2.5)

Equation (2.5) describes the variance of the depth estimation $\Delta L$ of the 3D scanner in terms of the variances of the angular resolution, of the camera $\sigma_\varphi$, and the projector $\sigma_\psi$. This equation holds not only for depth from structured
light scanners, but also for depth from stereo systems. In the latter case, $\sigma_\phi$ and $\sigma_\psi$ correspond to the angular resolutions of the two cameras.

Equation (2.5) applies to depth from structured light scanners with arbitrary patterns. The projector’s angular resolution $\sigma_\psi$ should be estimated for each specific design of a pattern, and substituted into (2.5) to obtain $\Delta L$.

For gray level patterns, the variance along the projector angle $\sigma_\psi$ can be expressed via the accuracy of the projection and the registration of the pattern intensity $\sigma_I$: $\sigma_\psi = \frac{\partial \psi}{\partial I} \sigma_I$. Thus, for gray level patterns, Equation (2.5) can be rewritten as

$$\Delta L = \frac{L^2}{d} \sqrt{\frac{\sigma_\phi^2}{\cos^4(\phi)} + \left(\frac{\partial \psi}{\partial I}\right)^2 \frac{\sigma_I^2}{\cos^4(\psi)}}. \tag{2.6}$$

Accordingly, the depth accuracy $\Delta L$ of a structured light 3D scanner is defined by the angular resolution of the camera $\sigma_\phi$, the intensity projection and registration accuracy $\sigma_I$, and the pattern profile, expressed in terms of
The rest of this chapter is devoted to the evaluation of (2.6) by the estimation of $\sigma_\psi$, $\sigma_I$, and $\partial \psi / \partial I$ based on the characteristics of the 3D scanner. In Section 2.3 we briefly describe a Complementary Metal Oxide Semiconductor Image Sensor (CMOS IS), and show how its accuracy, noise, and dynamic range are related to its design and physical principles of operation. In Section 2.4 we describe the mechanisms of image blur, which limits the camera angular resolution $\sigma_\varphi$. In Section 2.5 we discuss the influence of the pattern projector on the system accuracy. In Section 2.6 we discuss the influence of the object reflectivity and motion as well as the influence of the ambient illumination on the system accuracy. Finally, in Section 2.7 we provide accuracy estimates for some published results, and compare these estimates to the reported accuracies.

### 2.3 The Image Sensor

Equation (2.6) shows that an uncertainty in the image intensity $\sigma_I$ is translated into uncertainty in the distance $\Delta L$. Therefore, an accuracy of the image sensor is one of the crucial factors for the accuracy of structured light 3D scanners.

Two types of image sensors are used in modern cameras: Charge Coupled Devices (CCD) [58], [59], and Complementary Metal Oxide Semiconductor Image Sensors (CMOS IS) [60], [59].

CCDs are manufactured in a dedicated semiconductor process. A CCD sensor accumulates the light-generated electrons near the semiconductor surface, under an array of positively charged metal plates. The number of electrons, stored in each pixel, is proportional to the light exposure. After
exposure, the accumulated charges are shifted from pixel to pixel by sequential switching of their voltages, and transferred out of the chip for digitizing and processing [58].

CMOS image sensors are manufactured using a standard CMOS process, the same process used for manufacturing CPUs and other VLSI integrated circuits. CMOS image sensors have a photodiode as a sensitive element and a buffer amplifier for each pixel. Often the pixel sensor includes an A/D converter and other driving and processing circuitry integrated on the same chip. CMOS image sensors have an advantage over CCDs in terms of a higher integration level, higher speed, lower consumed power, enhanced dynamic range, and flexible pixel geometry.

Here we restrict our discussion to CMOS image sensors; however, most of the analysis is also applicable to CCD sensors.

2.3.1 Electrical scheme and operation principles of CMOS pixels

The architecture of a single CMOS IS pixel is shown in Figure 2.2.

The light sensitive element of an image sensor is a photodiode, denoted by $PD$. Before an acquisition of a new image, a reset transistor $Res$ opens for a short time to establish a reverse charge on the photodiode. This reverse charge closes the photodiode, hence the charge remains on the photodiode after closing of the reset transistor $Res$. The light absorbed during exposure generates electron-hole pairs, causing a reverse photo current through the closed photodiode, decreasing its charge [59].

At the end of the exposure, row and column selection transistors $Row$ and $Col$ are opened, and the residual voltage stored at the photodiode is
Figure 2.2: Scheme of a single pixel in a typical CMOS Image Sensor.

sensed and amplified by a buffer transistor $Buf$ and directed to an analog to digital converter, where it is digitized into a pixel value.

Ideally, the pixel value should reflect the exact quantity of the photons absorbed by the pixel during the exposure. Practically, the operation range is limited by the pixel sensitivity and dynamic range, while the signals within the operation range are distorted by the sensor non-linearity and noise.

### 2.3.2 Sensitivity and dynamic range

An ideal pixel should convert every photon reaching its surface into an electron-hole pair, count every electron discharged from the photodiode, and provide the number of counted electrons as an output. In reality, only a fraction of the photons produce an electron-hole pair thereby generating a photocurrent. Some photons are reflected back from the surface of the image sensor, while others may miss the photodiode, which covers only a fraction
of the pixel area [61].

Therefore, a photon reaching the pixel has some probability $\eta_q < 1$ of generating a conducting pair. This probability is called the quantum efficiency, and it is defined as

$$\eta_q = \lim_{n_{ph} \to \infty} \frac{n_e}{n_{ph}},$$

where $n_{ph}$ is the number of photons reaching the pixel, and $n_e$ is the number of electrons, discharged from the photodiode.

Usually, a single electron will not change the pixel readings. The number of photons, corresponding to the change of the least significant bit of the pixel, can be estimated as

$$n_{ph}^{\text{min}} = \frac{C_{\text{pix}} V_{\text{max}}}{\eta_q e^2 N},$$

where $C_{\text{pix}}$ is the pixel capacitance, $V_{\text{max}}$ is the maximum voltage on the photodiode, $e = 1.6 \cdot 10^{-19} \text{C}$ is an electron charge, and $N$ is the number of bits of the pixel output.

In many cases the full electron capacity $N_e = C_{\text{pix}} V_{\text{max}}/e$ (maximum number of electrons that can be stored in the pixel) is given in the data sheet. In this case, the number of photons corresponding to a single bit is

$$n_{ph}^{\text{min}} = \frac{N_e}{\eta_q 2^N}.$$  

In addition to the light-induced photo current, there is a thermally-induced leakage through the closed photodiode. This leakage is called the dark current, and it competes with the photo-current at low illumination levels, limiting the sensitivity of the image sensor. The dark current strongly depends on the temperature [59]

$$j_{\text{dark}} = j_0 e^{-\frac{T-T_0}{\sigma T}}.$$

(2.10)
\( \sigma_T \approx 11K \) in typical cases, indicating that the dark current doubles for every \( \sim 8K \). This means that cooling an image sensor by a mere \( 80K \) increases its sensitivity by a factor of \( \sim 1000 \) [59].

In order to reduce the effect of the dark current, some image sensors are designed with several additional rows of pixels, optically shielded by a layer of metal. These pixels can be used as a dark current reference, in order to estimate and subtract the dark current from the signal of the optically active pixels. This approach reduces the dark current effect, but does not eliminate it completely, since there is an uncertainty in the dark current value of each specific pixel, due to the shot noise and fixed pattern noise of the dark current [62], [63].

If the illumination is high enough, and the exposure is long enough, the charge stored in the photodiode is depleted completely during the exposure. This illumination corresponds to the saturation (maximum) value. Any illumination above the saturation level will not increase the pixel readings.

Usually, image sensors have 8-12 bits of resolution, corresponding to the dynamic range of 48-72dB. In order to increase the pixel dynamic range, a specially designed pixel with negative feedback on the reset transistor has been proposed [64]. In such sensors, the reset transistor gradually opens during the pixel discharge, partially recharging the discharged pixel. Such image sensors are often called logarithmic image sensors, because the feedback is designed so that the pixel response is roughly logarithmic with respect to the illumination level. For such architectures, dynamic ranges as high as 132 dB have been reported [65].
2.3.3 Noise and distortions

The ideal response of a pixel is linearly proportional to the illumination. Consistent and repeatable deviations from the linearity are referred to as distortions, while random fluctuations of the readings are referred to as noise.

The distortions are caused by the non-linearity of the voltage to the charge ratio of the photodiode, the non-linearity of the buffer transistor and the A/D converter.

Several different types of noise are usually distinguished: *Fixed pattern noise* is a consistent and repeatable difference in the sensitivity, offset, and dark current between the different pixels of the array. It is named fixed pattern noise due to the fact that the pattern of these variations over an image sensor array remains the same at each exposure. Fixed pattern noise can be significantly reduced by sensor calibration, when a frame measuring the dark current, and a frame measuring the sensitivity of the sensor array are stored in permanent memory and used to normalize each image frame.

The number of electrons $n_e$ generated in the pixel obeys the Poisson probability distribution [66], with an expected value $n_e = \eta_q n_{ph}$, defined by the quantum efficiency $\eta_q$ (2.7) and the number of photons $n_{ph}$ that reached the pixel. The variance of the number of electrons, $\sigma_{n_e} = \sqrt{n_e}$ is called the *shot noise*.

The ideal response of the pixel is an output signal, linearly proportional to the exposure. However, due to the quantization of the pixel output, a continuous increment of the exposure results in abrupt step-like changes in the response. The minimal step of the pixel reading is $\Delta I = I_{max}2^{-N}$, where $N$ is the number of bits. An intensity change corresponding to the half of the minimal step will either result in no change of reading, or in a change of $I_{max}2^{-N}$, corresponding to the least significant bit. In both cases the round-
off error will be $-2^{-(N+1)}$. Therefore, intensity quantization is equivalent to the noise, $\sigma_I = 2^{-(N+1)}$. This noise is called a quantization noise.

Quantization noise is the simplest to estimate, and for most of the image sensors it can serve as an estimate of the sensor noise, since the other types of noise are lower or of the same order of magnitude.

### 2.4 Image blur

From (2.5), one can see that the angular resolution of the camera $\sigma_\varphi$ is a crucial factor in defining the system accuracy. The angular resolution of the camera is defined by an image blur.

Formally, an image blur can be described as a convolution of a sharp pre-image $s(x)$ with an optical blur kernel $\mu(x)$, a defocus blur kernel $\eta(x)$, and a sampling blur kernel $\rho(x)$ [28], [67], [68]

$$h(x) = \int \int \rho(x-s)\eta(s-t)\mu(t)dsdt. \quad (2.11)$$

The optical kernel $\mu(x)$ can be estimated by the minimal physically achievable blur of the optics, which is the diffraction blur (the blur of a good lens in the center of image plane at the higher end of the aperture $F$ is comparable to the diffraction blur),

$$\mu(x) = \left[\frac{2J_1(\gamma)}{\gamma}\right]^2 \quad (2.12)$$

[3]. Here, $\gamma = (\pi|x|D)/(\lambda l)$, $J_1$ is a first Bessel function [31], $\lambda = 0.7 \cdot 10^{-6} m$ is the wavelength of the light, $l$ is the distance from the lens to the image plane, and $D$ is the lens diameter. In practical cases $l \approx f$, where $f$ is the focal length of the lens. Using the definition of the $F$-number, $F \equiv f/D$, we can write $\gamma \approx (\pi|x|)/(\lambda F)$.
The defocus kernel has a cylindrical shape given by

\[ \eta(x) = \begin{cases} \frac{1}{\pi r_d^2} & |x| \leq r_d \\ 0, & |x| > r_d \end{cases} \]  \hspace{1cm} (2.13)

where the radius of the cylinder is given by [67], [68]

\[ r_d = \left| \frac{\Delta L D}{l} \right| = \left| \frac{\partial l}{\partial L} \frac{\Delta L D}{l} \right| = \left| \frac{f^2 \Delta LD}{L^2} \right| \approx \left| \frac{f^2 \Delta L}{L^2 F} \right|. \hspace{1cm} (2.14) \]

Here, \( \Delta L \) is the distance from the object point to the plane in focus, the derivative \( \partial l/\partial L \), is obtained from the thin lens formula [4], connecting between the focal length \( f \), the distance \( L \) to the object, and the distance \( l \) to the image

\[ \frac{1}{l} + \frac{1}{L} = \frac{1}{f}. \hspace{1cm} (2.15) \]

Finally, the sampling kernel describes the averaging over a square pixel of size \( \Delta x \)

\[ \rho(x) = \begin{cases} \frac{1}{\Delta x^2} \max(|x_1|, |x_2|) \leq \frac{\Delta x}{2} \\ 0, & \text{otherwise.} \end{cases} \hspace{1cm} (2.16) \]

The convolution (2.11) is not integrable in elementary functions; however, an overall blur can be estimated as a square root sum of its components

\[ \sigma_x = \sqrt{\left( k \frac{\lambda F}{\pi} \right)^2 + \left( f^2 \frac{\Delta L}{L^2 F} \right)^2 + \left( \frac{\Delta x}{2} \right)^2}. \hspace{1cm} (2.17) \]

Finally,

\[ \sigma_\varphi = \frac{\sigma_x}{f} = \sqrt{\left( k \frac{\lambda F}{\pi} \right)^2 + \left( f^2 \frac{\Delta L}{L^2 F} \right)^2 + \left( \frac{\Delta x}{2} \right)^2}. \hspace{1cm} (2.18) \]

In many cases, the primary factor of the image blur is the sampling blur, in which case \( \sigma_\varphi = \frac{\Delta x}{2f} \).
2.5 Pattern projector

Various types of pattern projectors have been proposed. Due to space limitations, we restrict the discussion to projection with a video projector [69]. Nonetheless, the discussed problems are relevant to the other types of projectors.

A commercial video projector is the simplest way to generate structured light patterns without special hardware. A typical video projector consists of a halogen lamp, rotating color filter wheel, and a liquid crystal display or an array of moving micro-mirrors, manufactured using Micro-Electro-Mechanical-Systems (MEMS) technology. Such a micro-mirror array is called Digital Light Processing (DLP) technology, or a Spatial Light Modulator (SLM).

When generating a pattern with a video projector, a pattern is created in the computer and transmitted to the video projector. The real pattern generated by the video projector differs from the digital pattern in the computer, as described below.

2.5.1 Non-linearity and non-monotonicity

In the video projector, the pixel value goes through the gamma correction[70], $I_D = I_S^\gamma$, where $I_D$ is the physical value of the video projector output, $I_S$ is the value of the digital signal transmitted from the computer to the projector, and $\gamma \sim 2.2$ is the gamma factor. For structured light 3D scanners the pattern should have an exact intensity profile. Therefore, the gamma expansion of the projector should be compensated for by complementary gamma-compression of the signal.

The physical output of a commercial video projector should be treated
as an unknown non-linear function of its input video signal. Projector calibration is a necessary step prior to using the projector-generated patterns. The calibration can be done with either a linear ramp or a sequence of gray patterns of rising intensity, projected onto the white screen. Recording a signal with a camera (whose response is generally much more linear than the projector response) allows the measurement of the projector’s response function, and compensation for the signal by the reciprocal transformation of the generated pattern.

2.5.2 Pattern non-uniformity

The projected pattern is non-uniform in space due to the variance between the responses of different pixels, non-uniformity of the light source, and the projector optics.

The variance of the pixel responses can be rectified by defocusing the projector – most of the gray level patterns are invariant to defocus. The projector’s non-uniformity results in the projected patterns being brighter in the center and darker on the periphery. However, the patterns are invariant to multiplication by an arbitrary factor; therefore, the optics’ non-uniformity should not change the depth reconstruction.

2.5.3 Temporal noise

The temporal noise of the light projector results in variability of the pixel intensity between the patterns. This noise is translated into distortions in the reconstructed depth. If the temporal noise of a video projector is uncorrelated between the pixels, it can be decreased by defocusing the projector; otherwise, it can be decreased by time averaging of several patterns.
2.6 Ambient illumination, reflectivity and motion

2.6.1 Ambient illumination

An uncontrolled ambient illumination has several adverse effects on the performance of a 3D scanner. The patterns should be designed such that they are invariant to ambient illumination. Even in this case, additional ambient illumination decreases the effective dynamic range of the pattern, and a variation of the ambient illumination between the patterns results in distortions.

First, let us consider the set of two patterns \( \{p_1, p_2\} \):

\[
\begin{align*}
p_1 &= I_0 \frac{\psi + \Psi}{2\psi} \\
p_2 &= I_0,
\end{align*}
\]

where \( I_0 \) is the maximum projector intensity, \( \psi \) is an angle from the projector axis, and \( \Psi \) is the maximum projector angle in either direction from the optical axis (see Figure 2.1). This pattern set is invariant to the object albedo and distance, which is verified by multiplication of the patterns by an arbitrary factor \( \alpha \): \( \{p_1, p_2\} \mapsto \{\alpha p_1, \alpha p_2\} \)

\[
\psi = \Psi \frac{2p_1 - p_2}{p_2} = \Psi \frac{2\alpha p_1 - \alpha p_2}{\alpha p_2}.
\]

Now consider the case, when an ambient illumination \( \beta \) is added to the pattern: \( \{p_1, p_2\} \mapsto \{\alpha(p_1 + \beta), \alpha(p_2 + \beta)\} \). Then the depth is estimated by

\[
\Psi \frac{2\alpha(p_1 + \beta) - \alpha(p_2 + \beta)}{\alpha(p_2 + \beta)} = \psi + (\Psi - \psi) \frac{\beta}{\alpha(I_0 + \beta)}.
\]

The second term corresponds to the error in \( \psi \), caused by an ambient illumination. Its contribution to the error in \( L \) can be calculated by substituting \( \sigma_\psi = (\Psi - \psi) \frac{\beta}{\alpha(I_0 + \beta)} \) into (2.5).
In order to overcome the influence of ambient illumination, we consider the same patterns, with an additional third pattern

\[
\begin{align*}
    p_1 &= I_0 \frac{\psi + \Psi}{2\Psi} \\
    p_2 &= I_0 \\
    p_3 &= 0,
\end{align*}
\]

and the reconstruction formula \( \psi = \Psi \frac{p_1 - p_2 - p_3}{p_2 - p_3} \). This pattern set is invariant under change of object albedo and ambient illumination \( \{p_1, p_2, p_3\} \mapsto \{\alpha(p_1 + \beta), \alpha(p_2 + \beta), \alpha(p_3 + \beta)\} \):

\[
\Psi \frac{2\alpha(p_1 + \beta) - \alpha(p_2 + \beta) - \alpha\beta}{\alpha(p_2 + \beta) - \alpha\beta} = \Psi \frac{2\alpha p_1 - \alpha p_2}{\alpha p_2} = \psi.
\]

Despite the fact that this pattern set is invariant to ambient illumination, ambient illumination still decreases the accuracy of the 3D scanner. Consider a strong ambient illumination: \( \beta \approx I_0 \). Let us assume that the camera exposure has to be reduced by half, in order to adjust the dynamic range, so that the pattern and the ambient illumination together do not drive the camera into saturation. In this case, the derivative \( \frac{\partial\psi}{\partial I} \) doubles, and therefore, the angular error of the projector doubles in (2.6).

Another problem is that the ambient illumination may vary between the exposures in an uncontrollable manner. In this case there is no algorithmic solution to estimating the change in ambient illumination. Accordingly, \( \Delta L \) is estimated by (2.6), where the change in ambient illumination between the exposures is represented by \( \sigma_I \).

### 2.6.2 Object reflectivity and motion

The variation of object reflectivity increases the variation of the captured image intensity and therefore decreases an available dynamic range for the
pattern. For example, a well-known problem in the automotive manufacturing industry requiring 3D scanners is a bin-picking of flat metal-sheet parts. The reflectivity of these shining metal parts depends strongly on the incidence angle. The ratio between the highest and the lowest reflectivities can exceed 30 dB [71], [72]. This means that the darkest regions will be below camera sensitivity, and at the same time the brightest regions will be above the saturation level, and therefore, the 3D map for the whole part cannot be reconstructed.

Motion between frames blurs the object’s 3D shape. Let us consider the problem of 3D face recognition of the passengers, walking through a departure gate at the airport. The speed of walk of a pedestrian is about \( V_p = 1 \text{m/sec} \). Let us assume that the required accuracy is \( \Delta L = 0.5 \text{ cm} \), and that the scanner requires three frames for depth reconstruction. The required frame rate will hence be \( \nu = \frac{1 \text{m/sec} \cdot 3 \text{frames}}{0.05 \text{m}} = 600 \text{frames/sec} \), which shows the challenge in 3D acquisition of moving objects.

### 2.7 Revisiting depth from structured light

Zhang and Huang in [57] described a 3D scanner system, consisting of a digital Kodak DP900 video projector and a high speed Dalsa CA-D6-0512 camera. The camera resolution was 532 × 500 pixels, the projector repetition frequency was 80 Hz, and each cycle consisted of three color patterns.

They used three sinusoidal phase-shifted patterns [73]:

\[
\begin{align*}
    p_1 &= I_0 \sin(n\psi - \frac{2\pi}{3}) \\
    p_2 &= I_0 \sin(n\psi) \\
    p_3 &= I_0 \sin(n\psi + \frac{2\pi}{3}).
\end{align*}
\]
For these patterns,

$$\psi = \frac{1}{n} \arctan \left( \frac{2p_2 - p_1 - p_3}{\sqrt{3}(p_1 - p_3)} \right), \quad (2.25)$$

which can be verified by applying trigonometric identities. These patterns are invariant to object reflectivity and ambient illumination. This can be verified by substituting $$I_i = \alpha P_i + \beta$$ into (2.25), where $$\alpha$$ is the object reflectivity, and $$\beta$$ is the ambient illumination.

Now, consider

$$\sigma_\psi = \frac{1}{n} \left( \frac{\partial \psi}{\partial I_1} \sigma_{I_1} + \frac{\partial \psi}{\partial I_2} \sigma_{I_2} + \frac{\partial \psi}{\partial I_3} \sigma_{I_3} \right); \quad (2.26)$$

therefore,

$$\sigma_\psi = \left( \frac{2\sqrt{3}}{n} \right) \left( \frac{(I_3 - I_2)\sigma_{I_1} + (I_1 - I_3)\sigma_{I_2} + (I_2 - I_1)\sigma_{I_3}}{3(I_1 - I_3)^2 + (2I_2 - I_1 - I_3)^2} \right). \quad (2.27)$$

Substituting $$I_1 = I_0 \sin(n\psi - \frac{2\pi}{3})$$, $$I_2 = I_0 \sin(n\psi)$$, $$I_3 = I_0 \sin(n\psi + \frac{2\pi}{3})$$, and assuming uncorrelated and equal variances of the three images $$|\sigma_{I_1}| = |\sigma_{I_2}| = |\sigma_{I_3}| = |\sigma_I|$$, we obtain:

$$\frac{\partial \psi}{\partial I} = \frac{\sqrt{2}}{n}. \quad (2.28)$$

We are now ready to estimate the system accuracy, using (2.6):

$$\Delta L = \frac{L^2}{D} \sqrt{\frac{\Delta \phi^2}{\cos^4(\phi)} + \left( \frac{\sqrt{2}}{n} \right)^2 \frac{\Delta I^2}{\cos^4(\psi)}}. \quad (2.29)$$

The reported rms error for the flat surface was 0.05 mm. Unfortunately, the basis $$D$$ between the camera and the projector, and the distance $$L$$ to the object were not reported. From the photograph of the 3D scanner and the reported size of the box, we estimate $$D \approx 200$$ mm, and from the size of
the measured surface, 260 × 244 mm, we estimate $L \approx 300$ mm. We assume that the angular resolution of the camera is defined by pixel size, and can be expressed via the camera angle of view $\Phi$ and the number of pixels $N_{pix}$,

$$\sigma_\varphi = \frac{\Phi}{2N_{pix}},$$

Substituting $\Phi \approx 1\text{rad}$ and $N_{pix} = 500$ in the above, we obtain $\Delta \varphi = 0.001$. Substituting $n = 16\text{ periods/radian}$, and assuming that the patterns utilize the full dynamic range of the 8 bit camera $\sigma_\psi = \frac{1}{2^{256}}$, we obtain $\sigma_\psi \approx \frac{1}{5600}$.

The first term of (2.29) corresponds to the inaccuracy of the camera angle, due to averaging over a pixel and the optics blur, while the second term corresponds to the inaccuracy of the pattern angle due to the noise in the image intensity. The second term is the dominant one for system accuracy, since the image smoothing, described by the first term, results in the smoothing of the distance map, while the image noise, described by the second term, corresponds to the noise of the distance map.

Substituting $L = 300$ mm, $D = 200$ mm, $\sigma_\psi = \frac{1}{5600}$ into (2.29), we obtain $\Delta L \approx 0.08$ mm, which is comparable to the reported accuracy of 0.05 mm.

### 2.8 Chapter summary

We estimated the depth accuracy of structured light 3D scanners, by analysis of the angular resolutions of the camera $\sigma_\varphi$ and the projector $\sigma_\psi$ (2.5). This equation holds for depth from structured light scanners, as well as depth from stereo systems. The angular resolution of the camera is expressed in terms of the optical parameters of the system and the pixel size of the image sensor (2.18). The angular resolution of the projector is estimated through the accuracy of image intensity acquisition (2.6), which is defined by the
image sensor accuracy and the noise. We provided a description of a CMOS
image sensor, and showed how its accuracy, noise, and dynamic range are
related to its design and physical principles of operation. Next we described
the mechanisms of image blur, which limits the camera angular resolution
$\sigma_\phi$. Then the influence of the pattern projector noise and distortions, as well
as the object reflectivity, motion, and ambient illumination on the system
accuracy were evaluated. Finally, we demonstrated the use of the developed
estimators, by applying them to the scanner reported in [57].
Part II

Algorithms: Image Binarization, Machine Learning, and Dimensionality Reduction
Chapter 3

Image Binarization

3.1 Overview

The problem of binarization of gray level images, acquired under nonuniform illumination is reconsidered. Yanowitz and Bruckstein proposed to use for image binarization an adaptive threshold surface, determined by interpolation of the image gray levels at points where the image gradient is high. The rationale is that high image gradient indicates probable object edges, and there the image values are between the object and the background gray levels. The threshold surface was determined by successive over-relaxation as the solution of a Laplace equation. This chapter, based on [74, 75] proposes a different method to determine an adaptive threshold surface. In this new method, inspired by multiresolution approximation, the threshold surface is constructed with considerably lower computational complexity and is smooth, yielding faster image binarizations and often better noise robustness.
3.2 Introduction

Let us consider the problem of separating objects from their background in a gray level image $I(x, y)$, where objects appear lighter (or darker) than the background. This can be done by constructing a threshold surface $T(x, y)$, and constructing a binarized image $B(x, y)$ by comparing the value of the image $I(x, y)$ with $T(x, y)$ at every pixel, via:

$$B(x, y) = \begin{cases} 
1 & \text{if } I(x, y) > T(x, y) \\
0 & \text{if } I(x, y) \leq T(x, y). 
\end{cases}$$

(3.1)

It is clear that a fixed value of the threshold surface $T(x, y) = \text{const}$ can not yield satisfactory binarization results for images obtained under nonuniform illumination and/or with a nonuniform background.

Chow and Kaneko in [76] were among the first researchers to suggest using adaptive threshold surfaces for binarization. In their method the image was divided into overlapping cells, and sub-histograms of gray levels in each cell were calculated. Sub-histograms judged to be bimodal were used to determine local threshold values for the corresponding cell centers, and the local thresholds were interpolated over the entire image to yield a threshold surface $T(x, y)$. This was certainly an improvement over fixed thresholding, since this method utilized some local information. However, the local information was implicitly blurred to the size of the cell, and this, obviously, could not be decreased too much.

Yanowitz and Bruckstein made a step forward in [77] by suggesting to construct a threshold surface by interpolating the image gray levels at points where the image gradient is high. Indeed, high image gradients indicate probable object edges, where the image gray levels are between the object and the background levels. The threshold surface was required to interpolate the
image gray levels at all support points and to satisfy the Laplace equation at non-edge pixels. The surface was determined by a successive over-relaxation method (SOR) [77], [78].

Trier and Tasto conducted a performance evaluation of fifteen binarization methods by comparing the performance of OCR system with respective binarization method as the first step [79]. The Yanowitz-Bruckstein (YB) method produced the best results with the Trier-Tasto method just slightly behind. After the addition of a ghost-elimination step from Yanowitz and Bruckstein method, the methods of Niblack [80], Eikvil-Taxt-Moen [81] and Bernsen [82] performed slightly better.

As will be shown later, the last three methods are not scale-invariant, and their performance is optimal only for some specific object size or requires the parameter tuning. The Yanowitz-Bruckstein method is scale invariant, however the computational complexity of successive over-relaxation method is expensive: $O(N^3)$ for an $N \times N$ image and the resulting binarization process is slow, especially for large images. Furthermore, the threshold surface tends to have sharp extremum at the support points, and this can degrade the binarization performance.

We here follow the approach of Yanowitz and Bruckstein and use image values at the support high gradient points to construct a threshold surface. However, we define a new threshold surface via a method inspired by multiresolution representation [83]. The new threshold surface is constructed as a sum of functions, formed by scaling and shifting of a given orignal function. This new threshold surface can be stored in two ways: as an array of coefficients $a_{ijk}$, or as a conventional threshold surface $T(x, y)$ which is obtained as a sum of scaled and shifted source functions, multiplied by appropriate coefficients $a_{ijk}$.
The threshold surface coefficients $a_{ijk}$ are determined in $O(P \log(N))$ time, where $P$ is the number of support points and $N^2$ is the image size. These coefficients can then be used to construct the threshold surface $T(x, y)$ over the entire image area $N^2$ in $O(N^2 \log(N))$ time or to construct the threshold surface over smaller region of the image of $M^2$ size in only $O(M^2 \log(N))$ time. Furthermore, the adaptive threshold surface can be made smooth over all the image domain.

The rest of this paper is organized as follows: Section 3.3 reviews the best performing methods according to Trier and Tacht evaluation [79]: Niblack [80], Eikvil-Tacht-Moen [81], Bernstein [82], and Yanowitz-Bruckstein [77]. Section 3.4 describes a proposed new method to construct a threshold surface. Section 3.5 describes the implementation of the surface computation. Section 3.6 presents some experimental results, comparing the speed and binarization performance of the proposed method with the methods of Niblack and Yanowitz-Bruckstein. Finally Section 3.7 summarizes this work with some concluding remarks.

3.3 Review of Binarization Methods

3.3.1 Niblack’s Method

The idea of this method is to set the threshold at each pixel, based on the local mean and local standard deviation. The threshold at pixel $(x,y)$ is calculated as

$$T(x, y) = m(x, y) + k \cdot s(x, y),$$  \hspace{1cm} (3.2)

where $m(x, y)$ and $s(x, y)$ are the sample mean and standard deviation values, respectively, in a local neighborhood of $(x,y)$. The size of the neighborhood
should be small enough to reflect the local illumination level and large enough to include both objects and the background. Trier and Taxt recommend to take $15 \times 15$ neighborhood and $k = -0.2$.

### 3.3.2 Eikvil-Taxt-Moen’s Method

The pixels inside a small window $S$ are thresholded on the basis of clustering of the pixels inside a larger concentric window $L$. $S$ and $L$ are sliding across the image in steps, equal to the size of $S$ [79],[81]. For all the pixels inside $L$, Otsu’s threshold $T$ [84] is calculated to divide the pixels into two classes. If the two estimated class means $\hat{\mu}_1$ and $\hat{\mu}_2$ are further apart than a pre-defined limit $l$

$$
\| \hat{\mu}_1 - \hat{\mu}_2 \| \geq \ell,
$$

then the pixels inside $S$ are binarized using the threshold $T$. Otherwise, all the pixels inside $S$ are prescribed to the class with the closest updated mean value. Trier and Taxt recommend $S = 3 \times 3$, $L = 15 \times 15$ and $\ell = 15$.

### 3.3.3 Bernsen’s Method

For each pixel $(x,y)$, the threshold $T(x,y) = (Z_{\text{low}} + Z_{\text{high}})/2$ is used, where $Z_{\text{low}}$ and $Z_{\text{high}}$ are the lowest and highest gray level pixel values in a square $r \times r$ neighborhood centered at $(x,y)$. If the contrast measure $C(x,y) = Z_{\text{high}} - Z_{\text{low}} < \ell$, then the neighborhood consists of only one class, that is assumed to be a background. Trier and Taxt recommend $r = 15$ and $\ell = 15$.

### 3.3.4 Yanowitz-Bruckstein’s Method

The essential steps YB binarization method [77] are the following:
1. Find the *support points* $\{p_i\}$ of the image $I(x,y)$, where the image gradient is higher than some threshold value $G_{th}$,

$$p_i = \{x_i, y_i\} : |\nabla I(x_i, y_i)| > G_{th}. \quad (3.4)$$

2. Find the threshold surface $T(x,y)$ that equals to the image value at the support points and satisfies the Laplace equation at the rest of the image points:

$$T(p_i) = I(p_i)$$

$$\nabla^2 T(x,y) = 0 \text{ if } \{x,y\} \in \Omega \setminus \{p_i\} \quad (3.5)$$

Here $\Omega$ is the set of all the image points. The solution of (3.5) is found by a relaxation method.

3. Determine the binarized image $B(x,y)$ according to (3.1).

These three steps are a simplification of the original method, made in order to discuss the essential steps without being lost in the details. The original method also included the following steps. A smoothing of the image before Step 1. The one-dimensional relaxation along the image boundary between the Steps 1 and 2 in order to use the obtained values as the Dirichlet boundary conditions for Step 2. Discarding of ‘ghost’ objects after Step 3, determined as the objects in the binarized image with relatively small gradients along the edge.

The smoothing of original image and discarding of ghost objects were omitted here, while the one dimensional relaxation along the boundary and use of the result as Dirichlet boundary condition was substituted by the use of Neumann boundary conditions in Step 3.

The SOR starts with an approximate solution $t(x,y)$, and numerical iterations take it to the unique solution $T(x,y)$ of the Laplace equation [77].
3.3.5 Analysis of the Binarization Methods

In order to make a goal-oriented evaluation of the binarization methods Trier and Tauxt built an experimental character recognition module. The binarization methods were applied to a hand-written hydrographic maps. Elliptic Fourier descriptors were extracted from the contour curve of the figures to form 12 dimensional features. Then the extracted features were used into the quadratic classifier [85], assuming multivariate Gaussian distributions for each of the ten digit classes.

According to the evaluation by Trier and Tauxt, the modified methods of Niblack, Eikvil-Tauxt-Moen, Bernsen, and the Yanowitz-Bruckstein’s method were ranked respectively to places 1, 2, 3 and 4. Obviously, this evaluation procedure could serve a good indicator for the performance of the binarization methods not only for the applications of recognition of hydrographic maps but also for other character recognition applications. However, the authors note that the generalization of the results to other application domains is not straightforward.

In the following paragraphs, we show that the methods of Niblack, Eikvil-Tauxt-Moen, and Bernsen are scale dependent, and will not work properly if the object sizes or the scale of the illumination uniformity vary significantly along the image. The threshold surface, constructed in the Yanowitz-Bruckstein method does not have explicit scale dependency. However, we shall show that the properties of this surface shade a doubt on its optimality for image binarization.

In the Niblack’s method, Eq. (3.2) \( T(x, y) = m(x, y) + k \cdot s(x, y) \) defines the threshold \( T \) inside the square of a fixed size, typically 15 × 15. Every such region is separated into an object and a background. Consider a completely white region, say, at the blank region of the page. The pixels will have some
mean \( m \) and standard deviation \( s \). Whatever the intensity distribution of the pixels, some pixels will necessarily fall below the threshold \( T \) defined by (3.2). Therefore, in every image region of size \( 15 \times 15 \) some pixels will be classified as objects and some as a background. This will be a misclassification for the images having regions of blank or objects of size \( 15 \times 15 \) or larger. The recommended value \( k = -0.2 \) can be considered as an incorporation of the prior knowledge and reflects the fact that more bright background than the dark objects is expected.

In the Eikvil-Tuxt-Moen's method the problem of single-class regions is treated somewhat better, since the condition \( \| \hat{\mu}_1 - \hat{\mu}_2 \| \geq \ell \), in (3.3) detects the cases of a single class in a region.

However, the existence of a 'magic' size \( L = 15 \times 15 \) makes the method scale dependent. Obviously, this scale is about the best compromise, at least for the case studied by Trier and Tuxt, however it can be too small for cases when the objects are large and too large for the cases when the illumination changes too fast along the image.

Bernsen's method is also scale dependent, as can be shown by applying similar arguments.

In the Yanowitz-Bruckstein’s method there is no explicit scale factor, and therefore this method is more appropriate for the general cases. However the price of constructing the threshold surface that depends on the entire image is high computational complexity. Really, every method that is limited to a fixed square size will scale linearly with the size of the image \( t = O(N^2) \). In the relaxation solution each iteration requires \( O(N^2) \) operations for \( N^2 \) grid points and there should be \( O(N) \) iterations to converge to a solution, therefore the method complexity is \( O(N^3) \) [77]. The solution of (3.5) can be found in just a \( O(N^2) \) time using multigrid methods [86]. However, it
will become clear from the following paragraph that not only the speed of computation but also the properties of the threshold surface can be improved.

The general form of the solution of the equation (3.5) in the continuum limit is

$$ \phi(x, y) = \psi(x, y) - \sum_{i=1}^{P} q_i \cdot \log(\sqrt{(x - x_i)^2 + (y - y_i)^2}), $$

where $\psi(x, y)$ is smooth and bounded function [87]. This solution has singularities at the support points. In the case of a problem discretized on a finite grid, the iterative solution of (3.5) will be finite, yet, it will have sharp extrema at the support points. These sharp extrema and especially the hanging ‘valleys’ between them can cause the unwanted ‘ghost’ objects in the binarized image. These ghost objects where eliminated in [77], however, it is preferable to get rid of them already by a careful construction of the threshold surface. To illustrate the sharp extremas at the support points and the hanging ‘valleys’ in between, Figure 3.1 shows a surface computed by SOR for 100 support points with random values in the range of 0..100. The support points were randomly scattered over a 128 × 128 grid.

Ideally, a good threshold surface should indicate the local illumination level, which is usually a smooth function of the coordinates. Moreover, the value of an image at a support point probably indicates the local illumination level in its vicinity and there is no reason that it will be a local extrema. Hence what actually happens to the threshold surface obtained by SOR solution of the Laplace equation is not what we would expect a good adaptive threshold surface to be. Therefore, it would be better not to put an interpolation constraint on the threshold surface, but to construct it as a smooth approximation of the support points thus making it robust to noisy outliers.
among the support points. The next section describes a new efficient way to construct such a threshold surface.

### 3.4 The New Threshold Surface

We propose to construct and represent the threshold surface as a sum of functions, obtained by scaling and shifting of a single source function, similar to what is done in wavelets or multiresolution representations [74]. In multiresolution representation [83] the coefficients are calculated on the basis of an original signal that is known a priori. In our case the complete threshold surface is not known in advance, but only its approximate values at the support points: $T(p_i) = I(p_i) \equiv v_i$. Here $p_i = \{x_i, y_i\}$ and $v_i = I(x_i, y_i)$ denote the $i$–th support point and its value. This section presents an efficient way to construct surfaces that interpolate and approximate image values at the support points $I(p_i)$. First an interpolation algorithm is presented. However, the interpolation surface obtained is discontinuous and cannot serve as a good threshold surface. Therefore, a small modification to the interpolation algorithm is presented, that results in a continuous and smooth approximation surface.

Let us consider a unit step source function, given by

$$G_{000}(x, y) = \begin{cases} 
1 & \text{if } (x, y) \in \Omega(I) \\
0 & \text{if } (x, y) \notin \Omega(I).
\end{cases} \tag{3.7}$$

Here $\Omega(I)$ denotes the set of all the image points. All the other functions we shall use are generated by downscaling of this source function and shifting the downscaled functions around the image plane and thus cover only part of the image:

$$G_{ijk}(x, y) = G_{000}(x \cdot 2^l - j, y \cdot 2^l - k), \tag{3.8}$$
where $l = 0, \ldots, \log_2(N)$ is a scale factor and $j, k \in \{0, \ldots, 2^l - 1\}$ are spatial shifts.

The threshold surface will be given by

$$T(x, y) = \sum_{l=0}^{\log_2(N)} \sum_{j,k=0}^{l} a_{ijk} G_{ijk}(x, y).$$  \hspace{1cm} (3.9)

### 3.4.1 Interpolation Algorithm

Let us introduce an algorithm to calculate the decomposition coefficients $a_{ijk}$ in order to obtain an interpolating surface $T(x, y)$ by (3.9), passing exactly through all the support points $T(p_i) = I(p_i)$.

The algorithm runs as follows:

1. The decomposition coefficient $a_{000}$ is set equal to the average of all the support points

$$a_{000} = \langle v_i^{(0)} \rangle = \frac{1}{P_{000}} \sum_{i=1}^{P_{000}} v_i^{(0)}.$$  \hspace{1cm} (3.10)

Here the first zero in index 000 refers to the 0-th resolution level, the following 00 refer to the only possible spatial position at this level. The support points $\{p_i\}_{i=1}^{P_{000}}$ are defined by (3.4) and $P_{000}$ is the total number of support points. After step 1 every support point $v_i^{(0)}$ is already approximated by the average $a_{000}$, so it remains only to interpolate the difference between the value of every support point and the average.

2. The values of the support points are updated as follows:

$$v_i^{(1)} = v_i^{(0)} - a_{000}.$$  \hspace{1cm} (3.11)

The quantities $v_i^{(1)}$ will be referred to as the first order residuals.
3. The image is divided into four cells, with corresponding indexes \( \{jk\} \) relating to the spatial position of the cell: \{00, 01, 10, 11\}. The average of the updated support points \( v_i^{(1)} \) of each cell \( jk \) is calculated to yield the appropriate decomposition coefficient \( a_{1jk} \):

\[
a_{1jk} = \frac{1}{P_{1jk}} \sum_{p_i \in S_{1jk}} v_i^{(1)}. \tag{3.12}
\]

Here \( p_i \in S_{1jk} \) denotes a support point \( p_i \) that belongs to the cell at the 1-st resolution level, situated at the \( (j,k) \) spatial position. \( P_{1jk} \) denotes the number of support points in this cell.

4. After step 3 the values of support points in each cell \( jk \) are approximated by \( a_{000} + a_{1jk} \), so their values are updated to be

\[
v_i^{(2)} = v_i - a_{000} - a_{1jk} = v_i^{(1)} - a_{1jk}. \tag{3.13}
\]

5. Steps 3 and 4 are repeated for successive resolution levels. At every resolution level \( (l - 1) \) each of the \( 4^{l-1} \) cells of this level is divided into four cells to yield \( 4^l \) cells at the resolution level \( l \). The coefficients \( a_{ljk} \) of the cells at level \( l \) at \( (j,k) \) spatial position are set to be equal to the average of the residual values of the support points, belonging to this cell:

\[
a_{ljk} = \frac{1}{P_{ljk}} \sum_{p_i \in S_{ljk}} v_i^{(l)}. \tag{3.14}
\]

Here \( p_i \in S_{ljk} \) denotes a support point \( p_i \) that belongs to the cell at level \( l \), placed at \( (j,k) \) spatial position. \( P_{ljk} \) denotes the number of support points in this cell. After calculation of the coefficients \( a_{ljk} \), the values of the support points are updated:

\[
v_i^{(l+1)} = v_i^{(l)} - a_{ljk}. \tag{3.15}
\]
6. The procedure ends at the highest resolution level \( L = \log_2(N) \), when the size of the cell equals to one pixel. At this step there is at most one support point in every cell \( jk \), with a residual value \( v^{(L)}_i \). The coefficient \( a_{Ljk} \) is set to \( a_{Ljk} = v^{(L)}_i \).

The threshold surface, constructed in accordance with equations (3.7-3.9) with the coefficients \( a_{ijk} \) obtained by the algorithm as described in steps 1-6, will be an interpolation surface of the support points \( \{p_i, I(p_i)\} \), i.e. it will pass through every support point. This can be proved by the following argument:

Consider some arbitrary support point \( p_i \). The value of the threshold surface at this point will be

\[
T(p_i) = \sum_{l=0}^{L} a_{ljk}.
\]  

(3.16)

Where the \( jk_l \) chooses at every level \( l \) the cell that contains the \( p_i \).

On the other hand the residual value \( v^{(L+1)}_i \) of the support point \( p_i \) equals to (step (6)):

\[
v^{(L+1)}_i = v^{(0)}_i - a_{000} - a_{1j1k1} - \ldots - a_{LjkL} = 0,
\]  

(3.17)

which can be rewritten as:

\[
v^{(0)}_i \equiv I(p_i) = a_{000} + a_{1j1k1} + \ldots + a_{LjkL}.
\]  

(3.18)

From (3.16) and (3.18) it follows that for an arbitrary support point \( p_i \), \( T(p_i) = I(p_i) \).

Figure 3.2 shows the interpolation surface, obtained by our method for the same set of support points that was used for the over-relaxation solution, shown in Figure 3.1.
Figure 3.1: Solution of the Laplace equation by the over-relaxation method

Figure 3.2: Interpolating surface, obtained by a new interpolation method.
3.4.2 Approximating Source Function

The method presented in the previous section yields a surface that interpolates the support points. However the obtained interpolation surface is discontinuous. In order to obtain an $n$-continuously differentiable approximation surface, the source function (3.7) must be substituted by $n$-times continuously differentiable function vanishing together with $n$ first derivatives at the boundary of its support.

In the practical case of finite grid it is enough to consider a source function having a value and derivatives small enough at the boundary. However there are three additional requirements from the source function: (approximation) it should have value close to 1 in the domain of its cell; (normalization) the integral of the source function over its support must be equal to the image area; (smoothness) it should decrease gracefully towards the boundary of its support. The first two requirements are necessary in order to build the threshold surface really approximating the support points and the third one in order to have it practically smooth.

As a compromise between these contradicting requirements we chose a source function with support $[-1, 2] \times [-1, 2]$, extending over the image area $[0, 1] \times [0, 1]$. Therefore the threshold surface (3.9) is constructed with scaled functions, overlapping at each resolution level. It was found empirically that the source (3.19) gave a good performance.

$$G_{000}(x, y) = \begin{cases} \frac{e^{-(x-\frac{1}{2})^4-(y-\frac{1}{2})^4}}{\int_{-1}^{1} \int_{-1}^{1} e^{-(x-\frac{1}{2})^4-(y-\frac{1}{2})^4}}}, & \text{if } \{x, y\} \in [-1, 2]^2 \\ 0, & \text{if } \{x, y\} \notin [-1, 2]^2. \end{cases} \quad (3.19)$$

The point $\{x, y\} = \{\frac{1}{2}, \frac{1}{2}\}$ is the center of the image, spanning over $[0, 1] \times [0, 1]$. Figure 3.3 shows the source function (3.19). The support points that will determine the decomposition coefficients lie in the central cell $[0, 1] \times$
where the source function (3.19) is practically flat. Eight periphery cells will overlap neighboring functions thus making the threshold surface smooth.

Figure 3.4 shows the smooth threshold surface, constructed with the source function (3.19) for the same set of support points that was used to construct the interpolated surfaces of Figures 3.1 and 3.2. The figures 3.1, 3.2, and 3.4 show the support points by vertical spikes. Some of the support points of Figure 3.4 are lying apart from the threshold surface. This is due to the fact that support points have random values for demonstration purposes and therefore the approximating surface passes far from some of the support points. In real cases, the neighboring support points usually have similar values and the approximation surface will be close to them.

The new threshold surface is smooth. It does not necessarily pass exactly through the support points however this is an advantage rather than disadvantage, because if several neighboring support points have substantially different and 'noisy' values this indicates either that the threshold surface is under-sampled by the support points or that there is some error or noise in their values. In both cases there is not enough information at the support points about the threshold surface and the best thing to do is probably to set the threshold surface somewhere in between, as done by the proposed approximation algorithm.

### 3.5 Implementation

The algorithm described in the previous section was implemented in Matlab. The subsections below describe the data structures and then the algorithm implementation.
Figure 3.3: The source function, given by (3.19).

Figure 3.4: An approximating surface, obtained with source function (3.19)
### 3.5.1 Data Structures

The basic data structures are two arrays:

The first array is called `coeffs` (Table 3.2), it stores the decomposition coefficients of the cells $a_{ijk}$ in the first row and the number of support points $P_{ijk}$ of this cell in the second. $a_{ijk}$ denotes the decomposition coefficient of the cell $ljk$, which is situated at the $(j,k)$ spatial position at the level $l$ of the resolution. $P_{ijk}$ stores the number of support points in this cell.

First column of `coeffs` stores the single coefficient of the lowest level $a_{000}$ and the total number of support points $P \equiv P_{000}$, following are four columns of coefficients of the first level ($a_{100}, \ldots, a_{111}$) and number of points in each of these cells ($P_{100}, \ldots, P_{111}$), etc.

Every support point belongs to one and only one cell $ljk$ at every resolution level $l$. There are $\log_2(N)$ different resolution levels, starting from single cell of size $N \times N$ at level 0 to $N^2$ cells of size $1 \times 1$ at level $\log_2(N)$.

The second array, called `pointarr` (Table 3.1), has $P$ columns and $1 + \log_2(N)$ rows. Every column of `pointarr` contains the current value of the support point $p_i^{(l)}$ in the first row, and the indices $ind_i$ in other rows. These indexes refer to the cells which contain $p_i$ at every level $l$: $coeffs[; ind_i] = [a_{ijk}; P_{ijk}]$.

Figure 3.5 shows an example of a point, which belongs to cell $000$ at level 0 (as every point does), cell $100$ at level 1, to cell cell $211$ at level 2 etc.. This point will contribute in the construction of the threshold function only through the coefficients $a_{000}, a_{100}, a_{211}, \ldots$. These coefficients are stored in the first row, columns 1, 2, 11... of array `coeffs` (Table 3.2). Therefore the column of `pointarr`, corresponding to this point will have values 1, 2, 11,... in its second, third, fourth... rows.
3.5.2 Algorithm description

1. Array points (Table 3.1) is created and gradually filled. Every column $i$ of this table contains value of the point $p_i$ in the first row. For every point $p_i$ a calculation is performed to determine to which cell $l_jd_kd_l$ it belongs at each level $l$, $l=0,\ldots, \log_2(N)$. The positions of these cells in the array coeffs (Table 3.2) are filled into rows 2, $\ldots, N$ of $i$-th column of pointarr, and simultaneously, for every encountered cell the counter of the points belonging to this cell is increased in the array coeffs. This requires $P \log_2(N)$ calculations of the cell index and $P \log_2(N)$ increments of the point counters (because everyone of $P$ support points entered into $\log_2(N)$ cells).

2. The coefficients $a_{ljk}$ in the array coeffs are calculated. $a_{000}$ is set to be an average value of all the points (3.10). After this the value of every point in points is updated: average value is subtracted from it (3.11).

3. Step 2 is repeated for a higher level: Every point contributes its current value to the cell it belongs to, this value is divided by the number of points which belong to the cell. After all the points of a given level have contributed their residual values to the cells, their values are updated: from each point belonging to cell $l_jk$ the value of $a_{ljk}$ is subtracted.

4. The threshold surface is built based on the coeffs and the basis function (3.19). This requires $O(N^2 \log_2(N))$ operations.

So an approximation surface for $P$ support points scattered over $N^2$ grid is determined as a set of coefficients using $O(P \log_2(N))$ operations and built explicitly using $O(N^2 \log_2(N))$ operations.
In the reconstruction phase, the virtual coefficients beyond the image boundary were created to effectively maintain Newman boundary conditions.

3.6 Experimental results

The three methods, YB with adaptive threshold surface obtained by SOR and the new one with adaptive threshold surface obtained by multiresolution approximation and the Niblack’s method were compared for speed and quality of binarization. The programs were implemented in MATLAB and ran on an IBM-Thinkpad-570 platform with 128MB RAM and a Pentium-II 366 MHz processor.

Four artificial black-white images were generated by simulating non-uniform illumination of the black and white pattern. This allowed to give a quantitative measure of the error of the binarization method. The error was calculated as a normalized $L^2$ distance between the binarized and the original B/W image. The post-processing step of ghost-elimination was omitted for all the methods. Figures 3.6 through 3.21 show the grey level images and the black-white images, reconstructed by the three binarization methods. Table 3.3 presents the runtimes and the binarization errors for each method. In the Niblack binarization method the value of $k$ from (3.2) was chosen to minimize the error, independently for each image. It was equal +0.8, -0.6, +0.05 and -0.2 respectively for the 'Squares', 'Text', 'Rectangles', and the 'Stars' test patterns.
Figure 3.5: Cell Hierarchy.

Table 3.1: Array `pointarr`. Column $i$ contains the indices of the cells containing $p_i$.

Table 3.2: Array `coeffs`. Contains decomposition coefficients $a_{ijk}$ and number of support points $P_{ijk}$ in the cell $ljk$.

Table 3.3: Comparison of the speeds and performance of the Yanowitz-Bruckstein (YB) and Multiresolution Approximation (MA) and Niblack binarization methods.
Figure 3.6: Gray level image of 'Squares'.

Figure 3.7: 'Squares' binarized with YB method.
Figure 3.8: 'Squares' binarized with Multiresolution Approximation method.

Figure 3.9: 'Squares' binarized with Niblack method.
Figure 3.10: Gray level image of 'Text'.

Figure 3.11: 'Text' binarized with YB method.
Figure 3.12: 'Text’ binarized with Multiresolution Approximation method.

Figure 3.13: 'Text’ binarized with Niblack method.
Figure 3.14: Gray level image of 'Rectangles'.

Figure 3.15: 'Rectangles' binarized with YB method.
Figure 3.16: 'Rectangles' binarized with Multiresolution Approximation.

Figure 3.17: 'Rectangles' binarized with Niblack method.
Figure 3.18: Gray level image of 'Stars'.

Figure 3.19: 'Stars' binarized with YB method.
For the 'Squares' test patterns the YB method gave the best results. The 'Text' test pattern reveals the definite superiority of the Niblack method for this important class of images. The 'Rectangles' test pattern was created by addition of 1% salt and pepper noise to the gray level image of a geometric series of rectangles. For this pattern the proposed method gave the best results. Finally, for the 'stars' test pattern the proposed method gave the best results again. The YB method had difficulties in the large regions without support points near the boundary, while the Niblack method gave perfect binarization for the objects of specific scale, and produced less impressive results for larger objects, as predicted in Section 3.3.5.

3.7 Chapter summary

We proposed a new way to construct a threshold surface in order to improve the Yanowitz-Bruckstein binarization method. The new threshold surface is constructed with considerably lower computational complexity and hence in much shorter time even for small images. The new method allows even more gain in speed in region-of-interest processing scenarios. The new threshold surface can be made smooth and by the nature of its construction should be similar to the local illumination level. These qualities allowed to expect a better visual performance of the binarization process. A binarization with the new threshold surface was compared to Yanowitz-Bruckstein and the Niblack methods on the set artificial images, with 4 representative cases presented and discussed here. Considering the experimental results it is apparent that there is no clear winner. The Niblack method was somewhat better for the structures like Text. Another advantage of the method is its speed and simplicity of implementation. However, Niblack method is scale dependent,
and that made it inferior to Yanowitz-Bruckstein and our methods on the non-text images, where objects of different scales appeared. For these images our method was comparable or better than the Yanowitz-Bruckstein method, while having a significant speed advantage.
Figure 3.21: 'Stars' binarized with Niblack method.
Chapter 4

Machine Learning

4.1 Overview

We consider the classification problem as a problem of approximating a given training set. This approximation is constructed in a multiresolution framework, and organized in a tree-structure. It allows efficient training and query, both in constant time per training point. The proposed method is efficient for low-dimensional classification and regression estimation problems with large data sets.

4.2 Introduction

In this chapter, which is based on [88], and [89] we study the utilization of multiresolution analysis for machine learning. For that goal, we first briefly review some of the milestones of both machine learning and multiresolution analysis.
4.2.1 Machine Learning

Consider two dependent random variables $\vec{X}$ and $Y$ with a joint probability distribution function $f_{\vec{X},Y}(\vec{x},y)$. Then, for any given $\vec{x} \in \vec{X}$ there is a distribution of $y$: $P(y|\vec{x})$. Therefore, the dependency of $y$ on $\vec{x}$ can not be fully described by a single valued function $M(\vec{x})$. A possible measure of the accuracy of such a description is the Risk Functional, defined by

$$R(M) = \int \int_{\vec{X},Y} |y - M(\vec{x})| f_{\vec{X},Y}(\vec{x},y) d\vec{x} dy. \quad (4.1)$$

Consider a finite set $T$ drawn from a process with a distribution $f_{\vec{X},Y}(\vec{x},y)$

$$T = \{\vec{x}_i, y_i\}_{i=1}^m. \quad (4.2)$$

Suppose that $T$ is the only information available about $f_{\vec{X},Y}$. The Machine Learning Problem can be defined as the problem of searching for a function $M_{opt}(\vec{x})$ that minimizes the risk functional $R(M)$

$$M_{opt} = \arg\min_M R(M). \quad (4.3)$$

This problem is ill-posed since the same $T$ can be a result of different $f_{\vec{X},Y}$ with different respective $M_{opt}(\vec{x})$.

The regression estimation problem is the problem (4.1-4.3) where $y$ obtains continuous values, while in the classification problem it obtains discrete values. In the rest of this chapter we consider only the classification problem, although our method can be also applied for regression estimation. It can be shown that $M(\vec{x})$, minimizing (4.1) is given by a Bayesian Classifier [85] which decision at point $\vec{x}$ is the class with the highest posterior probability.

For simplicity, consider a two-class classification problem. Let the value $y = 1$ correspond to the first class and the value $y = -1$ to the second. Denote the probability of the first class by $P(y = 1|\vec{x})$ by $w_1(\vec{x})$, and the probability
of the second class by \( P(y = -1|\vec{x}) \) by \( w_2(\vec{x}) \). Then, the classification decision of the Bayesian classifier is given by:

\[
M_{\text{Bayes}}(\vec{x}) = \begin{cases} 
1 & \text{if } w_1(\vec{x}) \geq w_2(\vec{x}) \\
-1 & \text{if } w_1(\vec{x}) < w_2(\vec{x})
\end{cases}
\] \hspace{1cm} (4.4)

Therefore, the classification decision at \( \vec{x} \) can be obtained from estimating of the probability density of each class and choosing the class with the highest probability.

### 4.2.2 Multiresolution Analysis

In multiresolution analysis signal processing starts from low-resolution, and then the resolution can be selectively refined, when necessary [83, 90].

A sequence of spaces \( \{V_l\}_{l \in \mathbb{Z}} \) is called a multiresolution [83], if the following properties are satisfied

\( (I) \) \( \ldots \subset V_{-1} \subset V_0 \subset V_1 \ldots \subset L^2(R) \)

\( (II) \) \( \bigcap_l V_l = \{0\}, \bigcup_l V_l = L^2(R) \)

\( (III) \) \( f(x) \in V_l \iff f(2x) \in V_{l+1} \)

\( (IV) \) \( f(x) \in V_0 \implies f(x-k) \in V_0, k \in \mathbb{Z} \) \hspace{1cm} (4.5)

\( (V) \) \( \exists \phi(x) \), called scaling function, such that

\( \{\phi(x-k), k \in \mathbb{Z}\} \) is an orthonormal basis of \( V_0 \)

Here, \( R \) is the space of reals, \( L^2(R) \) is the space of functions with finite 2-norm

\[
\int_{-\infty}^{\infty} V_i^2(x)dx < \infty,
\] \hspace{1cm} (4.6)

and \( \mathbb{Z} \) is the set of integers.

Consider the space \( W_l \), which is an orthogonal complement of \( V_l \) in \( V_{l+1} \):

\( V_{l+1} = V_l \bigoplus W_l \). Consider the function \( \psi(\vec{x}) \), called wavelet, forming an
orthonormal basis \( \{ \psi(x - k) \} \) in \( W_0 \). Then, \( \psi_{lk}(x) = \{2^{l/2}\psi(2^l x - k)\} \) is an orthonormal basis in \( W_l \), and thus the basis

\[
\{ \phi(x-k), 2^{l/2}\psi(2^l x-k) ; l \in \{0, \ldots, L_{\text{max}}\}, k \in \mathbb{Z} \} \quad (4.7)
\]

spans the space \( V_L \)

\[
V_0 \bigoplus W_0 \bigoplus W_1 \bigoplus \cdots \bigoplus W_{L-1} = V_L. \quad (4.8)
\]

Therefore, the projection of \( f(x) \) on a space \( V_L \) (called an approximation at resolution level \( L \)) can be written as

\[
f_L(x) = \sum_{k=-\infty}^{+\infty} \langle f(x), \phi_{0k}(x) \rangle \phi_{0k}(x) + \sum_{l=0}^{L} \sum_{k=-\infty}^{+\infty} \langle f(x), \psi_{lk}(x) \rangle \psi_{lk}(x). \quad (4.9)
\]

The scalar products \( \langle f(x), \psi_{lk}(x) \rangle \) are called decomposition coefficients and denoted by \( c_{lk} \). If \( f(x) \) has a bounded support \( x \in [0, 1] \), the sums over \( k \) are truncated

\[
f_L(x) = c_0 \phi_0(x) + \sum_{l=0}^{L} \sum_{k=0}^{2^l-1} c_{lk} \psi_{lk}(x). \quad (4.10)
\]

Multi-resolution analysis is usually applied to one or two dimensional signals for compression, denoising or feature extraction. The scaling function and wavelet are constructed to provide fast decay of the coefficients with increasing \( l \).

It can be proved [83], that if the wavelet \( \psi \) has \( p \) vanishing moments

\[
\int_{-\infty}^{+\infty} x^k \psi(x) dx = 0 \text{ for } 0 \leq k < p, \quad (4.11)
\]

and the function \( f(x) \) is \( \alpha \) times continuously differentiable \( f(x) \in C^\alpha, \alpha < p \), then the magnitude of decomposition coefficients decay with the rate \( l^{-\alpha} \);

\[
\exists A : |c_{lk}| < Al^{-\alpha}. \quad (4.12)
\]

95
This is the reason for choosing wavelets with greater \( p \). Wavelets of the same resolution level \( l \) that correspond to different spatial positions \( k \) can overlap. If the wavelet \( \psi(x) \) has a compact support \( K \), then for an arbitrary point \( x_0 \) at each scale \( l \), there are \( K \) wavelets \( \psi_{l+1}, \ldots, \psi_{l+K} \), whose support includes \( x_0 \). It can be shown [91], that the wavelet having \( p \) vanishing moments will have a support of size at least \( K \geq 2^p - 1 \).

In common applications of multiresolution analysis, wavelets with \( 2 - 3 \) or more vanishing moments (and therefore support of size 3-5 or more) are usually used. In Section 4.3 we show that this choice is inappropriate for machine learning, since a support of size \( K > 1 \) results in bad generalization and computational properties.

The wavelet basis can be extended into two or more dimensions in the following way. Let \( \{V^2_i\}_{i \in \mathbb{Z}} \) be a two space multiresolution, defined by \( V^2_i = V_i \otimes V_i \), where \( V_i \) is defined in (4.5). Let \( W^2_i \) be an orthogonal complement of \( V^2_i \) in \( V^2_{i+1} \):

\[
V^2_{i+1} = V^2_i \bigoplus W^2_i. \tag{4.13}
\]

Let \( \phi \) and \( \psi \) be the scaling function and wavelet in \( V_0 \). Then, the functions

\[
\begin{align*}
\psi^1 &= \phi(x_1)\psi(x_2), \\
\psi^2 &= \psi(x_1)\phi(x_2), \\
\psi^3 &= \psi(x_1)\psi(x_2);
\end{align*}
\]

scaled as

\[
\psi^i_{l,k} = 2^{1D/2}\psi^i(2^l x_1 - k_1, 2^l x_2 - k_2), \tag{4.15}
\]

where \( D = 2 \) is the space dimensionality, form an orthonormal basis in \( W^2_i \).
4.3 Multiresolution Approximation for Machine Learning

Wavelet analysis is often used for compression, denoising, and feature extraction in image processing. In image processing, the wavelet analysis is applied to two-dimensional signals, whose values are known in points on a regular grid.

In machine learning problems, unlike image processing, the dimensionality $D$ of the feature space is usually higher $D > 2$ and the function $y = f(\vec{x})$ is unknown, but has to be constructed from the finite sample $T$ of distribution $f_{\vec{X},Y}(\vec{x},y)$.

This difference has dramatic consequences. One can see, from generalization of (4.14-4.15) to $D$ dimensional space, that there are $2^D - 1$ different wavelet functions at the same spatial position $\vec{k}$. Moreover, if a 1D wavelet has a support of size $K$, then in $D$ dimensions each feature vector $\vec{x}_0$ falls within the support of $(K + 1)^D - 1$ different wavelets. In order to find the decomposition coefficients, there must be other $(K + 1)^D - 1$ training points at appropriate locations around $\vec{x}_0$.

In order to have the complete information for calculation of the coefficients at resolution level $l$, there must be $(K + 1)^{lD} - 1$ training points at the appropriate locations. It is impractical to expect that $T$ will contain this amount of homogeneously distributed training points.

4.3.1 Prior Work

Bernard, Mallat and Slotine applied wavelets to Machine Learning in [92]. Due to geometric considerations, they used the 3-band (tryadic) version of
Deslauriers-Dubuc interpolation process and the wavelet system

\[ \psi_{lk} = 3^{l/2} \psi(3^l x - k) \text{ for } x \in [0, 1]. \]  

(4.16)

The multi-dimensional basis is constructed as a generalization of (4.14-4.15).

The number of the candidate wavelets is maintained equal to the number of training points by applying the allocation procedure that adds a new wavelet for each new training point. The allocation procedure favors wavelets with centers closer to training point and lower resolution level. Then, a system of linear equations is solved to obtain the decomposition coefficients interpolating the training set.

Due to spacial locality of wavelets, the obtained linear system is sparse, and using previous solution for \( N - 1 \) training points, the update for the \( N \)th point is done in \( O(\log^2 N) \) time.

The generalization ability of this method is limited, due to the above arguments, and the fast scale decrease at rate \( 3^{-l} \) instead of \( 2^{-l} \).

For example, such an interpolation for a training set of 2 points equidistant from the center will divide the feature space into unequal volumes \( 1 - \frac{1}{3^D} \) and \( \frac{1}{3^D} \), while the correct partition should yield equal volumes of \( \frac{1}{2} \) each. This limits the dimensionality of application of this method. The examples, presented in [92] are indeed limited to \( D = 2 \).

In our approach we construct the probability density function as the multi-resolution sparse approximation/interpolation. This results in the good generalization properties, with training and query complexity is \( O(1) \). This allows to apply the method to the problems with \( D \) 10 and arbitrarily large training sets, with the training and query speeds actually limited by the reading time from the hard drive.

98
4.3.2 Our Approach

In the rest of this chapter, we assume that the training set is prescaled into the unit cube $\vec{x} \in [0, 1]^D$. This can be done in time linearly proportional to the size of the training set.

Due to the reasons, discussed at the beginning of this section, we choose the simplest scaling function, with the smallest possible support $K = 1$;

$$\phi_0(\vec{x}) = \begin{cases} 1 & \text{if } \vec{x} \in [0, 1] \\ 0 & \text{if } \vec{x} \notin [0, 1]. \end{cases}$$  \hfill (4.17)

The wavelet function, corresponding to this scaling function is the Haar wavelet [93];

$$\psi_0(\vec{x}) = \begin{cases} 1 & \text{if } \vec{x} \in [0, \frac{1}{2}] \\ -1 & \text{if } \vec{x} \in (\frac{1}{2}, 1], \\ 0 & \text{if } \vec{x} \notin [0, 1]. \end{cases}$$  \hfill (4.18)

For this choice, a new training point falls within the support of a single scaling function $\phi$ at every level $l$. However, there are still $2^D - 1$ wavelets (4.14) and therefore the system of equations, defining the wavelet decomposition coefficients is still (severely) undetermined.

The scaling functions

$$\phi_k(\vec{x}) = 2^{l/2} \phi(2^l x - k), k \in \mathbb{Z}$$  \hfill (4.19)

form an orthonormal basis in $V_l$ (4.5). This allows to use only the basis of $\phi_k(\vec{x})$, without $\psi_k(\vec{x})$ to represent a function $f(\vec{x})$ at any resolution level $l$. Such a representation is somewhat redundant, since the information about approximation of $f(\vec{x})$ at level $l$ is derivable from an approximation of $f(\vec{x})$ at level $l+1$. However, this overhead of order $2^{-D}$ is justified by the generalization properties and computational efficiency. Therefore, in our approach, the
approximations \( f_l(\vec{x}) \) are constructed at several resolution levels \( l = 0, \ldots, L \), using the basis of scaling functions \( \{ \phi_{lk}(\vec{x}) \} \).

In the case of two class classification, let the first class have the values \( y_1 = 1 \), while the second \( y_2 = -1 \). The decomposition coefficients are calculated to approximate the training set values

\[
    c_{lk} = \frac{1}{n} \sum_{\vec{x}_i \in C_{lk}} y_i. \quad (4.20)
\]

Where \( \vec{x}_i \in C_{lk} \) means that the training point belongs to the cell \( C_{lk} \), and \( n \) is the total number of the points in that cell. One can see, that in the limit of infinite number of the points within the cell and infinitely small cell size \( (l \to \infty) \), the cell value corresponds to the probability density of the classes

\[
    \lim_{n,l \to \infty} \text{sign}(c_{lk}) = \begin{cases} 
        1 & \text{if } w_1(\vec{x}) \geq w_2(\vec{x}) \\
        -1 & \text{if } w_1(\vec{x}) < w_2(\vec{x}).
    \end{cases} \quad (4.21)
\]

Here, \( w_1(\vec{x}) \) and \( w_2(\vec{x}) \) denote respectively the probability density of the first and second classes. This coincides with the decision of the Bayesian Classifier (4.4).

However, in practice, there is neither need nor the data for the resolution level \( l \) beyond \( L_{\text{max}} = 5 \sim 10 \), which is confirmed by experiments presented in Section 4.4. This is explained by the observation that, for example, in ten dimensional feature space \( (D = 10) \), at the fifth resolution level \( (l = 5) \) there are \( 2^5 \cdot D = 2^{50} \) cells, spanning the unit cube \([0, 1]^D\).

The classification consists of training and query phases, which can be arbitrarily interleaved. At the training phase the decomposition coefficients are updated to interpolate (approximate) the training set. If there are cells at \( l = L_{\text{max}} \), containing the training points of different values, the training set is approximated by the average in these cells, otherwise it is interpolated.
exactly. There are \( L_{\text{max}} + 1 \) coefficients that are updated by each new training point, since there is one cell at each resolution level \( l = 0, \ldots, L_{\text{max}} \), containing that point.

The classification decision \( y(\vec{x}_0) \) for a query \( \vec{x}_0 \) is calculated as the sign of the value of the smallest available cell containing \( \vec{x}_0 \). This cell can be at \( l < L_{\text{max}} \).

The structural risk minimization principle [94] can be embedded into the classification decision, when among the available values of the cells from \( 0 \leq l \leq L_{\text{max}} \), the sign of the cell with the minimum structural risk yields the classification decision. Here, the empirical risk is the ratio \( \frac{n_1}{n_1 + n_2} \) between the number of points of dominating class within the cell \( n_1 \) and the total number of points \( n_1 + n_2 \). The VC dimension of a single cell is one, since it can be in one of the two states: \( c_{lk} \geq 0 \) or \( c_{lk} < 0 \).

The support of \( \phi_{lk}(\vec{x}) \) forms a rectangular grid of cells of size \( 2^{-l} \). Most of these cells at high resolution levels will be empty. These empty cells are identical zeros and need not to be stored. The cell at level \( l \) is divided into \( 2^D \) cells at level \( l + 1 \). Only non-empty cells at level \( l \) can have inside non-empty cells at level \( l + 1 \). This allows efficient organization of the non-empty cells into a tree structure, where non-zero cells point to their non-zero sons. The search or addition of a new cell in this structure requires \( O(D \cdot L_{\text{max}}) \) operations.

The classifier is implemented by two procedures: \texttt{LearnPoint()} and \texttt{Query()}. The first procedure implements learning of a single training point by updating the decomposition coefficients. The second procedure implements query, calculating the function value at a query point from the decomposition coefficients.
Tree Structure of the Decomposition Coefficients

In order to efficiently access the sparse coefficients, they are organized in a tree-structure $T$. Each node in the tree corresponds to the cell at level $l$ and includes pointers to the non-empty son cells at level $l + 1$; the height of the tree is $L_{\text{max}}$. The root of the tree corresponds to the largest cell $C_0$. Each node stores the number of points inside its cell $pnum$, the average value of these points $val$, the resolution level $l$, and the tree $sonptrs$ of pointers to its non-empty sons at level $l + 1$:

```c
struct treenode {
    float val; int pnum, level; sonstree *sonptrs;
}
```

Figure 4.1 shows a two dimensional feature space divided into cells at the first three resolution levels ($l = 0, 1, 2$). Figure 4.2 shows the organization of these cells into a tree structure.

Learning and Query Algorithms

The learning algorithm runs as follows:

Procedure $\text{LearnPoint}(tn, \vec{x}, y)$ receives three arguments. The structure $tn$ is a vertex of the tree $T$. Items $\vec{x}$ and $y$ are the coordinate and the value of the training point to be learned. The value of the cell $tn \rightarrow val$ is updated to be the average of all the points in the cell including the new one.

If the resolution level $tn \rightarrow level$ is lower than the maximum $L_{\text{max}}$, then the procedure $\text{GetSon}(tn, \vec{x})$ (not described here) returns a pointer to the corresponding son cell. If the cell does not exist in the tree it is created. The procedure is repeated for the son cell.

**Procedure** $\text{LearnPoint}(tn, \vec{x}, y)$

\% Increment number of points in the cell
Figure 4.1: Multiresolution representation of a 2-dimensional feature space.

Figure 4.2: Tree structure of the cells of a 2-dimensional feature space.

\[(tn \rightarrow pnum) = (tn \rightarrow pnum) + 1\]
\[\text{\% Update the cell value}\]
\[(tn \rightarrow val) = (tn \rightarrow val) + \frac{y - (tn \rightarrow val)}{tn \rightarrow pnum}\]
if \((tn \rightarrow level) < L_{\text{max}}\)
\[\text{\% Find or create the son cell}\]
\[sn = \text{GetSon1}(tn, \vec{x})\]
\[\text{\% Update the son cell}\]
\[
\text{LearnPoint}(sn, \vec{x}, y)\]
endif
return

The query algorithm returns the value of the smallest non-empty cell that contains the query.

**Procedure** Query\((tn, \vec{x})\)

\[sn = \text{GetSon2}(tn, \vec{x})\]
if \((sn \neq \text{NULL})\) then
\[\text{\% Repeat the Query for son cell}\]
return \textit{Query}(sn, \bar{x})

else

\% Return the value

return \textit{tn} \rightarrow \textit{val}

endif

return

4.3.3 Analysis

\textbf{Training time.} The procedure \textit{GetSon1}() returns a pointer to the unique son (among up to \(2^D\) cells), that contains the training point, in \(O(\lg(2^D)) = O(D)\) operations. There are \(L_{\text{max}}\) recursive iterations for \(L_{\text{max}}\) resolution levels, therefore, the overall complexity of the training is \(O(L_{\text{max}} \cdot D)\) operations per training point.

\textbf{Memory requirement.} In the worst case, every training point occupies \(L_{\text{max}} - 1\) independent cells. In this case, the memory needed for storage of the tree is \(O(L_{\text{max}} \cdot |T|) = O(|T|)\), where \(|T|\) is the size of the training set. In the case of a redundant training set it can be essentially lower. A pruning algorithm can be implemented, to control the size of the tree.

\textbf{Query time.} \textit{Query}() calls the \textit{GetSon2}() procedure up to \(L_{\text{max}}\) times, \textit{GetSon2}() (not described here) returns a pointer to the relevant cell in \(O(D)\) operations. Therefore, the \textit{Query}() complexity is \(O(L_{\text{max}} \cdot D)\).

Due to simplicity of the algorithms and the data structures, all constants in the complexity analysis are small, which results in training and testing speeds of tens of thousands points per second, actually limited by the hard-drive I/O speed.

\textbf{Application Domains:} Our classification method is based on the con-
struction of the interpolation (or approximation, if $L_{max}$ was insufficient to distinguish between training points of different values) function over the domain $[0, 1]^D$. For adequate sampling of this domain, at least $|T| = O(2^D)$ training points are required. Yet, the informative sampling of $[0, 1]^D$ requires $|T| = \Omega(2^D)$. This requirement naturally limits the dimensionality of the problems to $D \lesssim 20$. Problems with larger dimensionality require preliminary dimensionality reduction.

**Generalization Performance:** In cases, where the goal is to obtain the highest classification performance from a small training set, the proposed classifier is definitely not the best choice. The reason for this is that in the proposed classifier, the boundaries between different classes can pass only along the cell edges, with the coordinates defined by the rectangular grid at some resolution $2^{-l}$.

In popular classifiers like $k$–nearest neighbors, most of the decision trees, SVM, Neural Networks, and some others the boundaries are adaptive and continuously depend on the positions of the training points. This allows fine tuning to the training set and consequently better performance for small training sets. However, in cases where large amount of data is available or processing time is limited, the speed advantage of multiresolution approximation can transform into the advantage in classification performance, as demonstrated in Section 4.4.

In order to quantify the advantage in speed vs. disadvantage in classification performance, the expected error of a classifier as a function of its training time must be estimated. Unfortunately, both theoretical and experimental investigations of this question are subjective. Given two different classifiers, it is often easy to construct an example for which one or the other has better performance.
For example our classifier would be probably unbeatable on the $D$-dimensional chess-board patterned data of size $(2^D)^D$. At the appropriate resolution level, the black and white fields will coincide with the cells. However, for linearly separable training set the linear Perceptron [95] would obviously be a better choice etc.

### 4.4 Experimental Results

The proposed method was implemented in VC++ 6.0 and run on ‘IBM PC 300 PL’ with 600MHz Pentium III processor and 256MB RAM for the first two examples, and 1.8 GHz Pentium 4(m) and 512 MB RAM for the Forset Covertype example.

Although the classification problems with huge training sets seems to be important family of problems, it was hard to find such training sets in public databases.

Our method was tested on the Pima Indians Diabetes dataset [96], a large artificial dataset generated with the DatGen program [97] and the Forest Cover Type data set. The results were compared to [2, 98, 1, 99, 100].

#### 4.4.1 Pima Indians Dataset

This is an eight dimensional dataset consisting of 768 training points. The training set was re-scaled into a unit cube by linear transformation $f_{i}^{k'} = (f_{i}^{k} - f_{min}^{k})/(f_{max}^{k} - f_{min}^{k})$, here $i$ runs on 768 training points and $k$ over 8 dimensions. The classification performance was tested with ‘leave-one-out’ cross-validation. The results for this dataset are shown in Table 4.1.

This training set is relatively small and cannot really benefit from the speed and memory efficiency of the proposed method. The training set size
$|T| = 768 - 1 = 767$ is comparable to the number of the cells at the first resolution level $2^D = 256 (T \sim 2^D)$. For such sparse cases we have developed a modified query procedure, \textit{SmoothQuery}().

The idea behind the \textit{SmoothQuery}() is very simple: in the case when there are no training points within the cell $C_{lk}$ at level $l$, the values of the neighboring cells at level $l$ give more information about $C_{lk}$ than the value of the parent cell at level $l - 1$. Use of the neighboring values is equivalent to use of the overlapping basis functions at the query phase. The values of the neighbors are taken with weight factor $w(|\tilde{x}_{lk} - \tilde{x}_{lk'}|)$. Here, $\tilde{x}_{lk}$ is the center of the cell containing the query and $\tilde{x}_{lk'}$ is the center of its neighbor. The weight factor $w(r)$ can be empirically chosen, similarly to the choice of kernel in SVM. There are $3^D - 1$ neighboring cells, of which only a fraction is non-empty. These cells can be found efficiently due to the tree-structure organization of the non-empty cells.

Table 4.1 presents classification performance for training and test, the training and query times (per training point) and the memory required for the storage of the tree of coefficients. The first column presents the results for the interpolation algorithm (procedure \textit{Query}()), while the second column for the modified approximation algorithm (procedure \textit{SmoothQuery}()).

The \textit{SmoothQuery}() procedure, adapted for the small training sets, returns the value based not only on the cell containing the query, but also on the values of the neighbor cells. One can see that approximation takes $10^3$ longer, since in the function evaluation not only the cells, containing the training point but also their neighbor cells are taken into account.

The classification performance of 76.16\% (with standard deviation 2\%) was achieved by \textit{SmoothQuery}() with the training time of $24.8 \cdot 10^{-6} \cdot 768 = 1.9 \cdot 10^{-2}$ sec. This can be compared to 77.3\% performance and $\sim 4$ sec equiv-
alent training time for C4.5 with Rules [101], which was the best according to [100].

<table>
<thead>
<tr>
<th>Option</th>
<th>Interpolation</th>
<th>Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training time (sec)</td>
<td>24.80 \cdot 10^{-6}</td>
<td>24.80 \cdot 10^{-6}</td>
</tr>
<tr>
<td>Query time (sec)</td>
<td>11.62 \cdot 10^{-6}</td>
<td>16.35 \cdot 10^{-3}</td>
</tr>
<tr>
<td>Memory required, (kbyte)</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Train performance (%)</td>
<td>100.00</td>
<td>77.60</td>
</tr>
<tr>
<td>Query performance (%)</td>
<td>70.5</td>
<td>76.16</td>
</tr>
</tbody>
</table>

Table 4.1: Experimental results for the Pima Indians dataset

4.4.2 Large Synthetic Dataset in 6D

Another example is the large artificially generated dataset in 6-dimensional feature space with up to 500 thousand training points. This dataset was generated with the DatGen program [97] using the same call as in Section 3.2.2 of [1]:

```
datgen -r1 -X0/100,R,O:0/100,R, O:0/100,R,O:0/100,R,O:0/100,R,O:0/200,R,O:0/200
-R2 -C2/4 -D2/5 -T10/60 -O5020000 -p -e0.15 .
```

We have chosen this Dataset, since it was presented in [1]. This work is relevant to us, since it also treats the classification problem as problem of constructing an approximation function, and also demonstrates the training time linearly scalable with the training set and capability to digest training sets of millions of points.

The classifier is constructed as

$$M_{SpGr}(\vec{x}) = \sum_{i=1}^{N} \alpha_i \phi_i(\vec{x}),$$

(4.22)

where $M_{SpGr}$ denotes the Sparse Grid classifier [1], and the functions $\phi_i$ are from Schauder basis.
The one-dimensional Schauder basis is defined as
\[
\psi(x) = \begin{cases} 
1 - |x| & \text{if } x \in [-1, 1] \\
0 & \text{if } x \notin [-1, 1], 
\end{cases}
\] (4.23)
with the basis functions constructed by
\[
\psi_{lk} = 2^{l/2} \psi(2^l x - k) \text{ for } x \in [0, 1].
\] (4.24)

The multi-dimensional Schauder basis is constructed as generalization of (4.14-4.15).

The decomposition coefficients \( \{\alpha_i\} \) of \( M_{SpGr}(\vec{x}) \) are found to minimize the functional
\[
R(M_{SpGr}) = \frac{1}{m} \sum_{i=1}^{m} (M_{SpGr}(\vec{x}_i) - y_i)^2 + \lambda \|\partial_{\vec{x}} M_{SpGr}\|_{L_2},
\] (4.25)
where \( m = |T| \) denotes the size of the training set, the first term corresponds to the interpolation of the training set and the second to the smoothness of the function, in accordance with classical regularization approach [102]. In order to compute the decomposition coefficients, the function representation (4.22) is substituted into (4.25), and since \( \{\alpha_i\} \) correspond to the minimum, the derivative of the obtained system with respect to \( \alpha_i \) is set to zero. Thus, the following matrix equation is obtained
\[
(\alpha C + B \cdot B^T)\vec{\alpha} = B\vec{y},
\] (4.26)
where \( C \) is \( N \times N \) matrix with entries \( C_{j,k} = m \cdot (\partial_{\vec{x}} \phi_j, \partial_{\vec{x}} \phi_k)_{L_2} \), indices \( j, k = 1, \ldots, N \), and \( B \) is a rectangular \( N \times m \) matrix with entries \( B_{j,i} = \phi_j \vec{x}_i \), \( i = 1, \ldots, m; j = 1, \ldots, N \). The vector \( \vec{y} \) contains the data \( y_i \) and is of length \( m \). The vector \( \vec{\alpha} \) contains the unknown coefficients \( \{\alpha_i\} \) and is of length \( N \).
The value of $\bar{\alpha}$ satisfying (4.26) is found by iterative method. Sparse Grids Classifier suffers from the curse of dimensionality, which is the exponential rise of the complexity with increase of the dimension $d$. For example parsing the feature space with a grid with edge size $2^{-n}$ results in $2^{nd}$ grid cells. Employing the sparse grids reduces the complexity to $d n^{d-1} 2^n = d 2^{n+(d-1)\log_2 n}$. The overall computation complexity for a training set of size $M$ is $O(Md 2^{n+(d-1)\log_2 n})$, which is still very demanding even at $d = 6$, as can be seen in Table 4.2.

Table 4.2 shows the results of [1] in the upper part, and the corresponding results obtained by our method [88], in the lower part.

<table>
<thead>
<tr>
<th>Ref.</th>
<th># of points</th>
<th>training correctness</th>
<th>testing correctness</th>
<th>runtime, sec</th>
<th>memory used</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level 1</td>
<td>$5 \cdot 10^4$</td>
<td>90.8</td>
<td>90.8</td>
<td>158</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$5 \cdot 10^5$</td>
<td>90.7</td>
<td>90.8</td>
<td>1570</td>
<td></td>
</tr>
<tr>
<td>Level 2</td>
<td>$5 \cdot 10^4$</td>
<td>91.9</td>
<td>91.5</td>
<td>1155</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$5 \cdot 10^5$</td>
<td>91.5</td>
<td>91.6</td>
<td>11219</td>
<td>250MB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>This Work</th>
<th># of points</th>
<th>training correctness (stdev)</th>
<th>testing correctness (stdev)</th>
<th>train time, sec</th>
<th>test time, sec</th>
<th>memory used</th>
</tr>
</thead>
<tbody>
<tr>
<td>[88]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Level 1</td>
<td>$5 \cdot 10^4$</td>
<td>86.2 (0.35)</td>
<td>86.2 (0.35)</td>
<td>0.35</td>
<td>0.25</td>
<td>3.1k</td>
</tr>
<tr>
<td></td>
<td>$5 \cdot 10^5$</td>
<td>86.4 (3.4)</td>
<td>86.5 (2.4)</td>
<td>3.4</td>
<td>2.4</td>
<td>3.1k</td>
</tr>
<tr>
<td>Level 2</td>
<td>$5 \cdot 10^4$</td>
<td>90.7 (0.75)</td>
<td>89.3 (0.45)</td>
<td>0.75</td>
<td>0.45</td>
<td>163k</td>
</tr>
<tr>
<td></td>
<td>$5 \cdot 10^5$</td>
<td>91.6 (9)</td>
<td>90.5 (4.5)</td>
<td>9</td>
<td>4.5</td>
<td>197k</td>
</tr>
<tr>
<td>Level 3</td>
<td>$5 \cdot 10^4$</td>
<td>95.4 (1.55)</td>
<td>88.8 (0.85)</td>
<td>1.55</td>
<td>0.85</td>
<td>1.53M</td>
</tr>
<tr>
<td></td>
<td>$5 \cdot 10^5$</td>
<td>98.2 (9.5)</td>
<td>90.6 (9.5)</td>
<td>16.5</td>
<td>9.5</td>
<td>5.40M</td>
</tr>
</tbody>
</table>

Table 4.2: Experimental results for a 6D dataset, comparison of [1] and this work
The ∼1% disadvantage in classification performance of our method has low statistical significance, since the training sets were independently generated, and we have found about ∼1% standard deviation between different batches. The essential advantage in run-time of our method can be explained by absence of (explicit) regularization procedure. The cells, averaging the values of the training points inside them actually serve as a regularization, without additional computational burden.

4.4.3 Covertype Data

The Forest Covertype data is one of the largest data sets from UCI repository [96]. This data set contains 581012 examples with 54 attributes and 7 target classes and represents the forest cover type for $30 \times 30$ meter cells [2]. The 54 attributes of this data set actually represent 12 features. First 10 are numeric cartographic attributes. Last 44 attributes represent two features: the soil type, one out of 40 different types, is represented by a single non-zero bit among 40 bits; the wilderness area, one out of four, is represented by a non-zero bit out of 4 bits.

The reported classification performance for this data set was 70.58% in [2] and ∼72% in [99]. This relatively low performance can be partially explained by the fact that this large training set was not utilized completely: In [2], only 11340 examples were used as a training set. Even for this training set the training time was as much as 45 hours (‘UNIX Sun Sparc workstation’ of unreported configuration), and choice of the best network architecture required 56 such runs. In [99], the training set contained up to 65536 examples, while the run time was not reported.

In [2] fully connected network with one hidden layer and backpropagation training was used. After experimental search for optimal network architec-
ture and training parameters, the network with 120 hidden nodes (network 54-120-7), learning rate 0.05 and momentum rate of 0.9 was chosen. The training was stopped when one of the following three criteria was satisfied (1) MSE on the validation set decreased below 0.05; (2) insignificant decrease rate of the validation MSE; (3) 1000 training epochs were done.

Table 4.3 presents the details of the classification results of [2]. The columns correspond to the predicted classes, while the rows correspond to the classes that the test points actually belonged to. Thus, 33 258 points were predicted to be class 3, from them 25 295 (76.06 %) indeed were class 3, and 5697 were class 2, etc. 33 594 points out of 565 892 test points were points of class 3, what constituted 5.94 %. The overall classification performance (70.58 %) is the number of the correctly classified points from all the classes, divided by the size of the test set.

<table>
<thead>
<tr>
<th>Predicted</th>
<th>Observed total</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>201 928</td>
</tr>
<tr>
<td>c2</td>
<td>50 871</td>
</tr>
<tr>
<td>c3</td>
<td>2</td>
</tr>
<tr>
<td>c4</td>
<td>0</td>
</tr>
<tr>
<td>c5</td>
<td>20</td>
</tr>
<tr>
<td>c6</td>
<td>0</td>
</tr>
<tr>
<td>c7</td>
<td>638</td>
</tr>
<tr>
<td>Rate %</td>
<td>74.48</td>
</tr>
</tbody>
</table>

Table 4.3: Forest cover type. Classification matrix from [2], network 54-120-7, overall performance 70.58%

Since our method can not deal with 54 dimensional feature space, the last 44 dimensions, corresponding to the soil type and wilderness area were reduced into 2 numerical features. It was done by simple mapping of $40 \times 4 =$
160 possibilities into first 160 cells of $16 \times 16$ array. Such a crude dimensionality reduction probably reduces the classification performance, however the obtained results are still superior to the reported state of the art [2, 99]. The feature vectors were re-scaled into a unit cube by linear transformation,

$$f_i^k' = \frac{(f_i^k - f_{\text{min}}^k)}{(f_{\text{max}}^k - f_{\text{min}}^k)}.$$

The applied algorithm was a basic interpolation algorithm described above, with only one difference: Each cell contained 7 counters, each counting the number of training points of the corresponding class within the cell. The classification decision was done in favor of the class with the maximum number of points. There were NO empirical tuning parameters except the predefined maximum resolution level $L_{\text{max}}$. Therefore, our algorithm can be considered as multiresolution density estimator, with the classification decision in favor of the class with the highest local density.

Table 4.4 shows the overall classification performance as a function of the training set size and the maximum resolution level. The training and query time are measured without the loading time from the hard drive, which was 18.4 sec for 581 012 points. The standard deviation of the performance, evaluated over 10 runs is given in the parenthesis next to the performance.

The distribution of the classes varies throughout the data set. Therefore, the training and test sets were formed by homogeneous (with respect to the point number) split of the data set.

The expected standard deviation, proportional to $1/\sqrt{N}$, where $N$ is the size of the data sets is low. The standard deviation for 10 runs (different random partitions of the data set into training and test) at MaxLev=5, training set size 256 000 was measured to be 0.2%. The method was implemented in C (MSDN VC++ 6.0, no special compiler options) and run under Windows XP on 1.8 GHz Pentium 4(mobile) with 512 MB RAM.
Table 4.4: Forest cover type. Training, test times and classification performance and its standard deviation for different training set sizes and values of maximum resolution level

Table 4.5 shows our results for MaxLev=5 and Training set size of 512,000. The meaning of the cells is same as in Table 4.4.3.

4.5 Chapter summary

We proposed a new classification method, based on multiresolution density estimation of the classes in the training set. The decision of the Bayesian Classifier for any query \( \vec{x} \) is the class with the highest probability density function (pdf) in \( \vec{x} \). Our classifier is based on the multiresolution density estimation, which converges to the underlying probability density functions, and therefore it is Bayesian [88]. The method provides efficient training and
<table>
<thead>
<tr>
<th>Predicted</th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>c4</th>
<th>c5</th>
<th>c6</th>
<th>c7</th>
<th>total</th>
<th>Observed ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>25</td>
<td>497</td>
<td>33</td>
<td>896</td>
<td>4</td>
<td>298</td>
<td>294</td>
<td>932</td>
<td>2 252</td>
</tr>
<tr>
<td>c1</td>
<td>22</td>
<td>425</td>
<td>2</td>
<td>716</td>
<td>5</td>
<td>0</td>
<td>8</td>
<td>169</td>
<td>25 352</td>
</tr>
<tr>
<td>c2</td>
<td>2 768</td>
<td>30 263</td>
<td>205</td>
<td>1</td>
<td>175</td>
<td>139</td>
<td>31</td>
<td>33 582</td>
<td>48 66</td>
</tr>
<tr>
<td>c3</td>
<td>5</td>
<td>286</td>
<td>3</td>
<td>583</td>
<td>48</td>
<td>21</td>
<td>271</td>
<td>0</td>
<td>4 214</td>
</tr>
<tr>
<td>c4</td>
<td>0</td>
<td>11</td>
<td>85</td>
<td>218</td>
<td>0</td>
<td>25</td>
<td>0</td>
<td>339</td>
<td>0 49</td>
</tr>
<tr>
<td>c5</td>
<td>55</td>
<td>302</td>
<td>15</td>
<td>0</td>
<td>697</td>
<td>4</td>
<td>0</td>
<td>1 073</td>
<td>1 55</td>
</tr>
<tr>
<td>c6</td>
<td>13</td>
<td>261</td>
<td>405</td>
<td>27</td>
<td>9</td>
<td>1 369</td>
<td>0</td>
<td>2 111</td>
<td>3 06</td>
</tr>
<tr>
<td>c7</td>
<td>231</td>
<td>57</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2 052</td>
<td>2 341</td>
<td>3 39</td>
</tr>
<tr>
<td>Rate %</td>
<td>87.95</td>
<td>89.28</td>
<td>83.36</td>
<td>74.15</td>
<td>74.79</td>
<td>91.12</td>
<td>69 012</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.5: Forest cover type. Classification matrix for Multiresolution Approximation. Training set of size 512 000, maximum resolution level 5. Overall performance 87.85 %

query, linearly proportional to the training set size. The training and query speeds are actually limited by the hard drive I/O speed. The experiments show, that such advantage in training speed can result in higher classification performance for prohibitively large training sets.

The proposed classification method has similarities to Tree Classifiers, Nearest Neighbor k-d trees, Parzen Windows, and Wavelet Interpolation Networks.

It differs from the Tree Classifiers [98, 103] or the Nearest Neighbors organized in k-d trees [104] in that these classifiers have data-driven partitioning of the feature space, that prohibits or complicates the on-line learning and often penalizes the training time. Our classifier uses multiresolution basis functions, which can be considered as the a partitioning of space, however, since the geometry of the partitioning is defined by the basis, the online
learning is done via simple and straightforward change of few decomposition coefficients.

Parzen windows [105] prescribe the classification decision according to the majority of training points in a cell. However, a Parzen window classifier does not converge to the Bayesian classifier due to the fixed window size. Moreover, the fixed window can be too large for some regions of the feature space and at the same time too small for others. Our classifier uses multiresolution windows, which automatically span the feature space densely in populated regions, and uses the appropriate larger size cells to calculate the value in regions where the training points are sparse. Therefore, the proposed method can be considered as a consistent (converges to Bayesian) ‘Multiresolution Parzen Windows’

Wavelet Interpolation Networks [92] is another related approach. However, due to the limitations described in Section 4.3 the applications there are limited to 3 dimensions.
Bibliography


117


[37] F. Deschenes, D. Ziou, and P. Fuchs. Simultaneous computation of defocus blur and apparent shifts in spatial domain. *Actes de 15th International Conference on Vision Interface*.


[87] R. Courant and D. Hilbert. Methods of Mathematical Physics. Inter-
science Publishers, 1953.

[88] I. Blayvas and R. Kimmel. Machine learning via multiresolution ap-
proximation. IEICE Trans. on Information Systems, 86-D(6):1172–

[89] I. Blayvas, F. Bruckstein, and R. Kimmel. Machine learning by con-
struction of a decision function. US Patent 6,886,008, April 26, 2005.

[90] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact 
image code. IEEE Transactions on Communications, 31(4):532–540, 
1983.


for hierarchical approximation. In SPIE 44th Annual Meeting, Denver, 
CO, 1999.


126


תקציר

חברון זה בנוי משניים。

החלק הראשון דן בניתוח דיוק של סורקים תלת-ממדים, ובחלק השני מתאר אלגוריתם המהווה אבן בסיס למספר סורקים, וההוא אלגוריתם הבינריזציה. את אלגוריתם הבינריזציה אשר בנינו במהלך המחקר ניתן להכליל, לרחיב ולתפלי בעיות מתחום הלמידה החישובית, והורדת מעדים. בהתחדשות הטקסט: החיבור מתחיל מנ擬ון פסיול של מערכות לשחזור עומק מפוקוס או דה-פוקוס ממנה, והתחלואה החישובה, והורדת מעדים. חשיבות האלגוריתם המבנה טרומות רוב האלגוריתמים אשר פורסמו בספרות היו. הנוסחאות שפותחו במסגרת המחקר מאפשרות לשער את הדיוק של מערכות נתונות, או לתכנן מערכות חדשות, המתאימות לדרישות דיוק נתונות. הנוסחאות אשר נ겐ן ממ lx△ תרומת האלגוריתם לתרומת האופטיקה, והזאת מתוך הנתונים האופטיים.

האנליזה המבוססת על השוואת הטשטושים שונים של התמונה. הטשטוש הכללי של התמונה הוא סכום של הטשטוש האופטי, הטשטוש החישוני וטשטוש של דיפוקוס. טשטוש האופטי הוא תלוי בטחונות העדשה ולא ניתן לספור במקרא כללי. אך חסם התחתון לגודל הטשטוש הוא טשטוש דיפראקציוני. הטשטוש של עדשות טובות ברמות גבוהות של סגירת הצמצם מתקבב לחסם דיפראקציוני. טשטוש של החיישן המינימלי הוא טשטוש אל שטח של הפיקסל הבודד של החיישן. שעוני מ瓊 של טשטוש החישון לוחב ב частности של החיישן הבינארי, שהוא מספר זוגות הטטרנגל בפיקהליים והיא תליוןstrument פסיכולוגיה של החיישן. טשטוש של דיפוקוס מתקיים בקירוב של אופטיקה ליניארית, אך נ değiştir מדגמם של מודל דיפראקציוני של דיפוקוס. בטקסט של חיבור זה נדון גם בבעיות אילו.

הניתוח המת santé מראה כי האופטיקה משחקת תפקידי חיוני בדיוק המערכת, והשתרומת רוב האלגוריתמים אשר פורסמו בספרות היו. הנוסחאות שפותחו במסגרת המחקר מאפשרות לשער את הדיוק של מערכות נתונות, או לתכנן מערכות חדשות, המתאימות לדרישות דיוק נתונות. הנוסחאות אשר נגן של אלגוריתם הבינריזציה, והזאת מתוך הנתונים האופטיים. מספר פרסומים חשובים בתחום הפרמטרים האופטיים חיוניים לניתוח, ולא ניתן למספר את יעיליותה אלגוריתמית של המערכות המוצגות במאמרים אלו ללא תלות באופטיקה. לאור זאת, תוצאות המחקר אינן מוגבלות בתחום הסורקים התלת-ממדים, אלה גם כפרסום לקהילת התוכנית הממוחשבת, לפיו האופטיקהعتبر פרמטר חשוב במערכות ראיה, ונתוני הממדים האופטיים חיים להיוות ממונחים, או לפחות ממונחים במאמר זה. האנליזה המבוססת על השוואת הטשטושים שונים של התמונה. הטשטוש הכללי של התמונה הוא סכום של טשטוש האופטי, טשטוש החישוני טשטוש של דיפוקוס. טשטוש האופטי הוא תלוי בטחונות העדשה ולא ניתן לספור במקרא כללי. אך חסם התחתון לגודל הטשטוש הוא טשטוש דיפראקציוני. הטשטוש של עדשות טובות ברמות גבוהות של סגירת הצמצם מתקבב לחסם דיפראקציוני. טשטוש של חיישן המינימלי הוא טשטוש אל שטח של הפיקסל הבודד של החיישן. שעוני מ瓊 של טשטוש החישון לוחב ב частности של החיישן הבינארי, שהוא מספר זוגות הטטרנגל בפיקהליים והיא תליוןstrument פסיכולוגיה של החיישן. טשטוש של דיפוקוס מתקיים בקירוב של אופטיקה ליניארית, אך נتغير מדגמם של מודל דיפראקציוני של דיפוקוס. בטקסט של חיבור זה נדון גם בבעיות אילו.

האנליזה המבוססת על השוואת הטשטושים שונים של התמונה. הטשטוש הכללי של התמונה הוא סכום של טשטוש האופטי, טשטוש החישוני טשטוש של דיפוקוס. טשטוש האופטי הוא תלוי בטחונות העדשה ולא ניתן לספור במקרא כללי. אך חסם התחתון לגודל הטשטוש הוא טשטוש דיפראקציוני. הטשטוש של עדשות טובות ברמות גבוהות של סגירת הצמצם מתקבב לחסם דיפראקציוני. טשטוש של חיישן המינימלי הוא טשטוש אל שטח של הפיקסל הבודד שלحياء. שעוני מ瓊 של טשטוש החישון לוחב ב частности של חיישן הבינארי, שהוא מספר זוגות הטטרנגל בפיקהליים והיא תליוןstrument פסיכולוגיה של חיישן. טשטוש של דיפוקוס מתקיים בקירוב של אופטיקה ליניארית, אך נتغير מדגמם של מודל דיפראקציוני של דיפוקוס. בטקסט של חיבור זה נדון גם בבעיות אילו.
The thesis discusses multi-resolution analysis and signal analysis, focusing on interpolation and wavelet analysis. The author highlights the importance of VC dimension in understanding the behavior of algorithms in high-dimensional spaces.

The core idea is to analyze the signal in different resolutions, allowing for more effective decision-making and interpolation. The thesis proposes a method where the signal is processed at multiple resolutions, with the highest resolution handling the most detailed information and lower resolutions dealing with coarser details.

The author also examines the VC dimension of the algorithm, which is crucial for understanding its ability to generalize from a finite sample to an infinite sample. This is important for ensuring that the algorithm performs well on unseen data.

Overall, the thesis provides a comprehensive analysis of multi-resolution signal analysis and wavelet analysis, emphasizing their practical applications and theoretical foundations.
The method of dimensions reduction is too large to be included in the mentor group. The method of optimal dimension reduction is not optimal for the test group. We call this phenomenon "excessive reduction of dimensions".

We demonstrate how regularization improves the process of dimension reduction. We start with a general framework of dimension reduction and define the principles. We then show how to present the Linear Discriminant Analysis and Generalized Singular Value Decomposition, which are used in cases where the number of dimensions increases with the number of points in the mentor group. We demonstrate the excessive use of algorithm LDA, and present regularized LDA as a solution to the problem.
על הדיווח של سورקים אופטיים תלת מימדיים, אלגוריתמים בינריזציה וlearner חישובית

혀בר על מחקר

לשםميلויחלכםשהחרישותלקבלהתואר
דוקטורלפילוסופיה

איליהבליזבר

הוגשלאונטטכנוגר–מנמנטיילנדליירל

שנתה’תשס”ט–יוליונובמבר2009
המחלק נגעשת בהנחיית פרופסור רון קימל בפקולטה להנדסת מחשבים.

אני מודה לטכניון אל התמיכה הכספית בהשתלמויות. אני מודה לפרופסור רון קימלgerät אפרוד ברוקשטיין על התמיכה והדרכה.