CODEN: LUTFD2/(TFRT-5397)/1-28/(1989)

Nonlinear interpolation from video images to high quality printers

Anders Hansson

Department of Automatic Control
Lund Institute of Technology
Februari 1989

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
Master thesis

Date of issue
February 1989

Document Number
CODEN:LUTFD2/(TFRT–5397)/1-28(1989)

Author(s)
Anders Hansson

Supervisor
Lars Nielsen, Gunnar Sparr

Sponsoring organisation

Title and subtitle
Nonlinear interpolation from video images to high quality printers

Abstract

Recent high quality printers pose interesting questions on how to reproduce images taken by a video camera. A modern ink jet plotter has typically a resolution four times better than a good video camera. Pixel replication and bilinear interpolation give reasonable results, but have a tendency to blur the image. In contrast to these blurring phenomena, it is well known that the apparent visual quality is crucially dependent on how well the singularities, like edges, are reproduced. It is therefore intriguing to base the interpolation on a theoretical formulation that explicitly is based on image singularities. Classical potential theory fulfills these requirements, and the aim has been to start investigating the ability of this theory to capture intrinsic image properties. Computational algorithms have been searched for in two ways. One direction is to use numerical methods on the full problem. Another direction is to search for formulations approximating the full theory but having less computational complexity. Algorithms of both types are developed. They are nonlinear and make use of an edge-detection technique that is implemented as a least-square-minimization. An explicit description of images by means of a small number of primitives has been obtained. The nonlinear interpolation schemes have shown to preserve edges well.

Key words
Image interpolation, Edge-detection, Potential theory, Ink jet plotter

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
28

Recipient's notes

Security classification

Preface

This Master thesis has been carried out at the Department of Automatic Control in Lund. I would like to thank everybody who have helped me. The following persons are especially mentioned for their contribution to my work.

I would like to thank my two main supervisors. Lars Nielsen for his encouraging support and guidance throughout the work. He has, in spite of a lot of other work, always been able to find time for discussions, for which I am very grateful. I am also very grateful to Gunnar Sparr, who has frequently given me many inspiring suggestions. The potential-theory approach is one of his ideas.

For plotting images on the ink jet plotter I have collaborated with the group of Hellmuth Hertz at the Department of Electrical Measurement, Lund. I would especially like to thank Johan Nilsson and Thomas Johansson for their patient assistance in this part of the work. Closely related to this are the problem of writing on tapes for the plotter, for which I would like to thank Leif Andersson. Here I would also like to mention Tomas Schönthal. Leif Andersson has also helped me with TEX. To grab video-images the assistance of Toni Eriksson has been very helpful. It was also he who had read about the error-diffusion-technique.

I am also indebted to Mats Lilja, Kjell Gustafsson, Steve Murphy and Ola Dahl. Without their encouraging help in answering questions about the UNIX-operating system, the C-programming-language and TEX, I would have had to spend much more time with my work. I would also like to thank Bo Bernhardsson for his helpful suggestions concerning mathematics.

Lund February 1989

Anders Hansson

Contents

1. Introduction

Image expansion and high resolution reconstruction are interesting research fields touching on interpolation, approximation and digitization. Recent high quality printers have resolutions that are say four times better than video quality, for instance ink jet plotters. The reproduction thus involves an interpolation where pixel replication and bilinear interpolation are current techniques. These methods give reasonable results, but have a tendency to blur the image. In contrast to these blurring phenomena, it is well known that the apparent visual quality is crucially dependent on how well the singularities, like edges are reproduced. It is therefore intriguing to base the interpolation on a theoretical formulation that explicitly is based on image singularities. Classical potential theory fulfills these requirements, and the aim here is to start investigating the ability of this theory to capture intrinsic image properties.

The research, where the first steps are presented here, aims both at a basic understanding and at efficient computational algorithms. These computational algorithms are searched for in two ways. One direction is to use numerical methods on the full problem. Another direction is to search for formulations approximating the full theory but having less computational complexity. Preliminary results in these directions are presented here. The following two chapters describe the hard- and software used and the basic concepts of image interpolation. Chapters 4 and 5 describe image interpolation based on potential theory and experimental results from different methods of interpolation.

2. Hard- and software

A conventional computer based image handling system has been used to obtain true video-images. More specifically an IBM-PC-AT extended with a video-camera and a Matrox PIP 1024B image-processing-board have been used (Eriksson and Nielsen, 1989). The format of an IBM-PC-AT video image is 1 byte per pixel and 512×512 pixels per image. The value 0 corresponds to black and 255 to white. The bulk of the computations have been performed at a SUN workstation, and the images have been plotted on an ink jet plotter constructed by the group of Hellmuth Hertz at the Department of Electrical Measurement, Lund (Björkengren et al, 1985).

2.1 Ink jet plotter

The ink jet plotter is a drum plotter, and it consists of four different units: the plotting unit with four electronically controlled ink jets plotting the colors magenta, yellow, cyan and black, a magnetic tape transport, a buffer memory and a control unit. The image to be plotted is read from a tape, on which the image is stored as one line per block. The linelength is 5600 bytes, and the maximum number of lines is 4000. Each pixel of an image corresponds to 2 bytes of a line. For each of the four colors 4 bits of the 2 bytes is reserved. When using only one color, only the 4 least significant bits are used. This makes it possible to represent 16 different intensities for each color.

There are two characteristic properties of the ink jet plotter that has to be considered. Different intensities are obtained by different number of ink drops plotted at each pixel. The correspondence between bit code, number of spots and intensity is shown in Table 2.1 and Figure 2.1. It is easely seen that the relationship between bit code and intensity is not linear.

Unfortunately 16 different intensities are not enough for the eye to satisfactory represent an image with continuous-valued intensity (Rosenfeld and Kak, 1982, p. I:106)—level curves will occur. One way to overcome this problem is to use a so called error-diffusion-technique. When translating from intensity to bit code, which is done from left to right and from top to bottom, let the error in intensity caused by translating a specific pixel be added to surrounding not yet translated pixels—3/8 of the error to the east neighbor, 3/8 to the south and 2/8 to the south-east. This will cause the intensity to be correct in a mean sense. It is important to measure the error in intensity and not in bit-code, since the relationship is not linear. The nonlinear transformation from bit code to apparent intensity in combination with error diffusion has been suggested and implemented during the present work. The results are quite satisfactory.

Bit-code	Number of drops	Intensity
0	0	0.000
1	1	0.068
2	2	0.134
3	3	0.206
4	4	0.288
5	5	0.378
6	6	0.468
7	7	0.558
8	8	0.642
9	9	0.722
10	10	0.784
11	12	0.848
12	14	0.892
13	16	0.914
14	19	0.956
15	22	1.000

Table 2.1 The correspondence between bit code, number of spots and intensity for the ink jet plotter

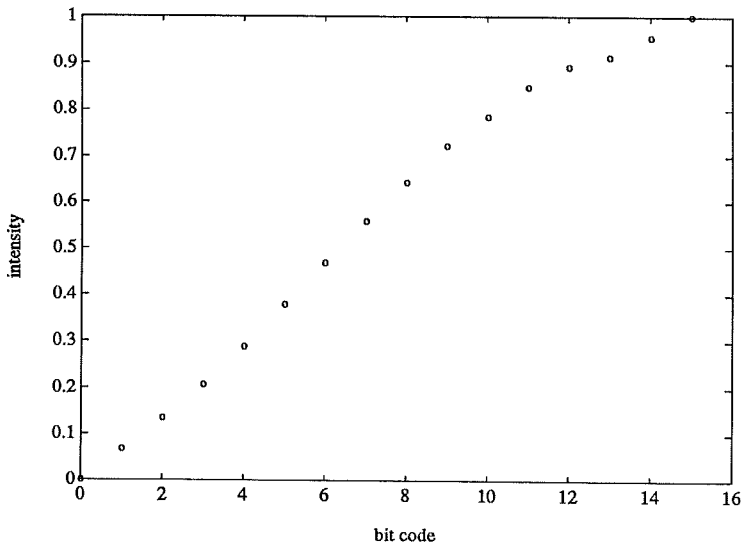

Figure 2.1 The correspondence between bit code and intensity for the ink jet plotter

2.2 Computational environment

The interpolation computations have been performed on a SUN workstation. A VAX computer with band-station has been used to write on tapes and obtaining correct block-length. The program on the SUN workstation is written in C (Kernighan and Ritchie 1978), and the program on the VAX computer in PASCAL.

Images at the SUN are stored as one float-number, 4 bytes, in the range $[0.0, 1.0]$ per pixel. Routines have been written to convert images between this representation and the representations used on the IBM and the ink jet

plotter.

The work at the SUN has been facilitated by writing routines to extract a coarse grid image from a fine grid image and to take out a rectangle of an image, to rotate an image 90°, to make elementary operations on an image such as $+$, $-$, $\cdot$, $/$, sin, cos, tan, log, ... , to compute convexity, to standardize an image that is out of the range $[0.0, 1.0]$, to display an image on a bilevel screen, to list numbers in an image, to convolve an image with an impulse response, to list file-names, to list file-headers, to list impulse response-files and to delete files.

The interpolation schemes described in the following chapters are implemented using these basic routines.

3. Basic concepts

Current techniques for image interpolation are mainly pixel replication and bilinear interpolation (Pratt, 1978). More sophisticated methods based on cubic polynomials have also been investigated (Bernstein, 1976; Rosenfeld and Kak, p. II:36).

In this chapter first some mathematical concepts are introduced. Images are represented by functions of two variables, which can be defined on continuous domains or grid domains. The ranges ar supposed continuous despite the fact that in computers no true real numbers exist. However for the purpose in Chapters 3 and 4 they will show to serve well, and the reader is referred to Chapters 2 and 5 and Appendix B for implementation details. It is also discussed how grid representations of images ("grid images") are obtained from continuous ones by means of continuous convolution in a video-camera and how—for evaluation purposes—a coarse grid may be approximately obtained from a fine d:o. Finally it is also shown that pixel replication and bilinear interpolation are linear operations, in the sense that they can be expressed as convolutions.

3.1 Image characterization

The following notations are introduced:

f	Original image
$\tilde{f}$	Filtered image
$\hat{f}$	Interpolated image
Ω	Definition set (Domain)
V	Value set (Range)
g	Index for "grid"
cg	Index for "coarse grid"
fg	Index for "fine grid"
L_{x_1}, L_{x_2}	Width and height
M, N	Resolution in coarse grid, i.e. $(M+1)(N+1)$ is the number of pixels in the coarse grid
R_{x_1}, R_{x_2}	Ratio between resolution in fine and coarse grid (typically $R = 4$)

$$\Omega = \{(x_1, x_2) \in \mathbb{R}^2 \mid 0 \leq x_i \leq L_{x_i}, \quad i = 1, 2\}$$
$$\Omega_{cg} = \left\{(\tfrac{x_1 L_{x_1}}{M}, \tfrac{x_2 L_{x_2}}{N}) \in \mathbb{Q}^2 \mid x_1 = 0, \ldots, M, \quad x_2 = 0, \ldots, N\right\}$$
$$\Omega_{fg} = \left\{(\tfrac{x_1 L_{x_1}}{R_{x_1} M}, \tfrac{x_2 L_{x_2}}{R_{x_2} N}) \in \mathbb{Q}^2 \mid x_1 = 0, \ldots, R_{x_1} M, \quad x_2 = 0, \ldots, R_{x_2} N\right\}$$
$$V = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$$
$$L_{x_1}, L_{x_2} \in \mathbb{R}^+$$
$$M, N, R_{x_1}, R_{x_2} \in \mathbb{N}$$

where Ω_{cg} and Ω_{fg} are rectangularely spaced grid points in Ω.

By means of sampling $\tilde{f}_{cg}$ is derived from f. Let the impulse response of a linear and invariant sampling filter be

$$h_{cg}(x) = \frac{1}{2\pi\sigma_{cg}^2} e^{-\frac{x_1^2+x_2^2}{2\sigma_{cg}^2}}, \quad x \in \mathbb{R}^2, \quad \sigma_{cg} \in \mathbb{R}^+.$$

A coarse grid image may then be derived as

$$\tilde{f}_{cg}(q) = f * h_{cg}(q), \quad q \in \Omega_{cg},$$

where

$$f * h_{cg}(x) = \int_\Omega f(y) h_{cg}(x-y)\, dy, \quad x \in \Omega$$

(Pratt, 1978,). By some sort of interpolation scheme an interpolated continuous image $\hat{f}$ may be obtained. This one can easily be sampled to obtain

$$\hat{f}_{fg}(q) = \hat{f}(q), \quad q \in \Omega_{fg}.$$

3.2 Evaluation

A possible way to evaluate an interpolation scheme is to compare $\hat{f}_{fg}$ with a sampled fine grid image obtained from f , e. g.

$$\tilde{f}_{fg}(q) = f * h_{fg}(q), \quad q \in \Omega_{fg},$$

where

$$f * h_{fg}(x) = \int_\Omega f(y) h_{fg}(x-y)\, dy, \quad x \in \Omega$$

and

$$h_{fg}(x) = \frac{1}{2\pi\sigma_{fg}^2} e^{-\frac{x_1^2+x_2^2}{2\sigma_{fg}^2}}, \quad x \in \mathbb{R}^2, \quad \sigma_{fg} \in \mathbb{R}^+.$$

Fortunately $\tilde{f}_{cg}$ can be obtained approximately from $\tilde{f}_{fg}$ as

$$\tilde{f}_{cg}(q) \approx \frac{L_{x_1} L_{x_2}}{2\pi M N R_{x_1} R_{x_2} (\sigma_{cg}^2 - \sigma_{fg}^2)} \sum_{p \in \Omega_{fg}} \tilde{f}_{fg}(p)\, e^{-\frac{(q_1-p_1)^2+(q_2-p_2)^2}{2(\sigma_{cg}^2-\sigma_{fg}^2)}},$$

$$q \in \Omega_{cg}$$

(see Appendix A), since otherwise two images at different distances would have had to be taken. What is actually done is that f is approximately reconstructed from f_{fg} with a piecewise constant function, which is then convolved with a coarse grid sampling-filter to obtain f_{cg}.

3.3 Pixel replication

The simplest form of interpolation scheme is pixel replication (Pratt, 1978), i. e. piecewise constant interpolation. It is linear, since it can be expressed as a convolution of $\tilde{f}_{cg}$ with

$$h(x) = \begin{cases} 1, & 0 \le x_1 \le \frac{L_{x_1}}{M}, \quad 0 \le x_2 \le \frac{L_{x_2}}{N} \\ 0, & \text{otherwise} \end{cases}, \qquad x \in \mathbb{R}^2.$$

Let $\text{ent}(x)$ = the greatest integer $\le x$. It then holds

$$\begin{aligned} \hat{f}(x) &= \int_\Omega \left(\sum_{q \in \Omega_{cg}} \delta(y-q)\, \tilde{f}_{cg}(y)\, h(x-y) \right) dy \\ &= \sum_{q \in \Omega_{cg}} \tilde{f}_{cg}(q)\, h(x-q) \\ &= \tilde{f}_{cg}\left(\frac{L_{x_1}}{M} \text{ent}\left(\frac{M x_1}{L_{x_1}} \right), \frac{L_{x_2}}{N} \text{ent}\left(\frac{N x_2}{L_{x_2}} \right) \right), \quad x \in \Omega. \end{aligned}$$

3.4 Bilinear interpolation

Another commonly used scheme for interpolation is bilinear interpolation (Pratt, 1978), which can be formulated as a convolution of $\tilde{f}_{cg}$ with

$$h(x) = \begin{cases} \left(1 - \frac{M}{L_{x_1}} x_1\right)\left(1 - \frac{N}{L_{x_2}} x_2\right), & 0 \le x_1 \le \frac{L_{x_1}}{M}, \quad 0 \le x_2 \le \frac{L_{x_2}}{N} \\ \left(1 + \frac{M}{L_{x_1}} x_1\right)\left(1 - \frac{N}{L_{x_2}} x_2\right), & -\frac{L_{x_1}}{M} \le x_1 \le 0, \quad 0 \le x_2 \le \frac{L_{x_2}}{N} \\ \left(1 + \frac{M}{L_{x_1}} x_1\right)\left(1 + \frac{N}{L_{x_2}} x_2\right), & -\frac{L_{x_1}}{M} \le x_1 \le 0, \quad -\frac{L_{x_2}}{N} \le x_2 \le 0 \\ \left(1 - \frac{M}{L_{x_1}} x_1\right)\left(1 + \frac{N}{L_{x_2}} x_2\right), & 0 \le x_1 \le \frac{L_{x_1}}{M}, \quad -\frac{L_{x_2}}{N} \le x_2 \le 0 \\ 0, & \text{otherwise} \end{cases}$$

$$x \in \mathbb{R}^2. \tag{3.1}$$

It holds

$$\begin{aligned} \hat{f}(x) &= \int_\Omega \left(\sum_{q \in \Omega_{cg}} \delta(y-q)\, \tilde{f}_{cg}(y)\, h(x-y) \right) dy \\ &= \sum_{q \in \Omega_{cg}} \tilde{f}_{cg}(q)\, h(x-q) \\ &= f_{00} + (f_{10} - f_{00})(x_1 - x_{1_0}) + (f_{01} - f_{00})(x_1 - x_{1_0}) \\ &\quad + (f_{00} - f_{10} - f_{01} + f_{11})(x_1 - x_{1_0})(x_1 - x_{1_0}), \quad x \in \Omega, \end{aligned}$$

where

$$
\begin{aligned}
x_{1_0} &= \frac{L_{x_1}}{M} \operatorname{ent}\left(\frac{M x_1}{L_{x_1}}\right) \\
x_{2_0} &= \frac{L_{x_2}}{N} \operatorname{ent}\left(\frac{N x_2}{L_{x_2}}\right) \\
f_{00} &= \tilde{f}_{cg}(x_{1_0}, x_{2_0}) \\
f_{10} &= \tilde{f}_{cg}(x_{1_0} + \frac{L_{x_1}}{M}, x_{2_0}) \\
f_{01} &= \tilde{f}_{cg}(x_{1_0}, x_{2_0} + \frac{L_{x_2}}{N}) \\
f_{11} &= \tilde{f}_{cg}(x_{1_0} + \frac{L_{x_1}}{M}, x_{2_0} + \frac{L_{x_2}}{N}).
\end{aligned}
$$

4. Edge-preserving interpolation

A disadvantage of the linear interpolation schemes in the previous chapter is the tendency of blurring when passing from a coarse to a fine grid. Of course it is possible to get a less blurred image, if a high-pass-filter is used, but this would cause an overall edge overshoot. It would be much more pleasing only to preserve edges where they are most likely to occur. Earlier results mainly treat bilevel images like text (Abdou and Wong, 1982). As will be seen below, edge-preservation can be achieved also for more complicated images by means of non-linear interpolation schemes based on potential theory.

4.1 Analytic description—electrostatic analogy

Suppose that f is twice continuously differentiable function in a regular domain Ω, except for at a finite number of smooth oriented curves γ_i, γ_i' (such a curve may degenerate to a point). By the normal direction of an oriented curve the right hand one is meant. Then, with

$$K(x) = -\frac{1}{2\pi}\ln|x|, \quad \frac{\partial}{\partial n} = \text{normal derivative} \quad \text{and} \quad \triangle = \frac{\partial}{\partial x_1^2} + \frac{\partial}{\partial x_2^2}$$

holds

$$f(x) = -\int_\Omega \{\triangle f(y)\}K(x-y)dy + \sum_i \int_{\gamma_i} \frac{\partial f}{\partial n}(y)K(x-y)ds_y + \sum_i \int_{\gamma_i'} f(y)\frac{\partial K}{\partial n}(x-y)ds_y$$

(Kellogg, 1954, p. 219 ff; Sparr, 1984, p. 5.7 f). In an electrostatic analogy, the first term is the potential caused by a continuous distribution of charges $\rho = -\{\triangle f\}$ (pointwise derivatives) outside γ_i, γ_i', the second term comes from curves γ_i with a single layer of charges with density $\alpha = \left(\frac{\partial f}{\partial n}\right)_\pm$ = jump across the curve, and the third from curves γ_i' with a double layer of charges (dipoles) with moment density $\beta = f_\pm$ in the direction of the right hand normal. When applied to a function f, representing an image, the formula describes how f is built up by means of elementary building blocks: *edges*, caused by the dipole lines; *shades*, caused by the charge lines and the *background* caused by the continuous charge distribution. For a justification of the terminology, see Figure 4.1 below.

Suppose that a piecewise continuous distribution of charges ρ, charged curves γ_i with density α and dipole curves γ_i' with moment density β are given. Given also with the boundary values of f, the potential f is then uniquely determined as the solution of a Poisson boundary value problem.

The same idea can be used for grid images f_g, where application of a discrete Laplacian gives a grid chart of the charge densities. If instead the charge

Figure 4.1 The results of a Poisson-solver on—from left to right—a single charge, a charge-line, a single dipole and a dipol-line

densities and the boundary values of f_g are given, then f_g can be recovered by means of a Poisson-solver.

The solutions in the cases of a single charge, a charge-line, a single dipole and a dipol-line are shown in Figure 4.1. The 1:st and 3:rd image show patterns that seem to be uncommon in most natural images. The 2:nd shows a shade and the 4:th an edge. Of these, the case of image 4, containing a discontinuity, suffers worst, when a conventional linear interpolation scheme is used. Guided by this fact the work is concentrated upon detecting dipole lines to preserve edges.

4.2 Edge-analysis

Given a coarse grid image $\tilde{f}_{cg}$, the charge densities ρ_{cg} are calculated as a convolution of $\tilde{f}_{cg}$ with the discrete Laplacian h as

$$\rho_{cg}(q) = -\sum_{p \in \Omega_{cg}} \tilde{f}_{cg}(p) h(q-p), \quad q \in \Omega_{cg},$$

where

$$\left(\frac{MN}{L_{x_1} L_{x_2}}\right)^2 h(q) = \begin{cases} -4, & q_1 = q_2 = 0 \\ 1, & \left(q_1 = 0 \wedge \mid q_2 \mid = \frac{L_{x_2}}{N}\right) \vee \left(\mid q_1 \mid = \frac{L_{x_1}}{M} \wedge q_2 = 0\right) \\ 0, & \text{otherwise} \end{cases}$$

(Rosendfeld and Kak, 1982). Let A be a region in Ω. Suppose that ($*$ = complement)

$$\tilde{f}(x) = \begin{cases} \beta, & x \in A \\ 0, & x \in A^* \end{cases}$$

and

$$\tilde{f}_{cg}(q) = \begin{cases} \beta, & q \in A \bigcap \Omega_{cg} \\ 0, & q \in A^* \bigcap \Omega_{cg} \end{cases}.$$

Then, with derivatives in distribution sense and n = outward normal from A, holds

$$\rho(x) = -\triangle \tilde{f}(x) = -\beta \left[\frac{\partial \delta}{\partial n}\right]_{\partial A}(x), \quad x \in A,$$

where $\left[\frac{\partial \delta}{\partial n}\right]_{\partial A}(x)$ denotes a unit dipole layer on the curve ∂A. The relation between $\tilde{f}_{cg}$ and ρ_{cg} is illustrated in Figure 4.2.

An interesting observation is that a knowledge of relative magnitudes of the values of ρ_{cg} at the vertices of a square crossed by an edge also contains information about the behavior of the edge in a neighborhood of the square.

$$\frac{\tilde{f}_{cg}}{\beta}$$

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

$$-\frac{\rho_{cg}}{\beta}$$

0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0
1	−2	−1	−2	2	0	0	0	0	0	0	0
1	−1	0	0	−2	2	1	1	0	0	0	0
1	−1	0	0	0	−1	−1	−2	1	0	0	0
1	−2	0	0	0	0	0	−1	2	0	0	0
0	2	−2	−1	0	0	0	0	−2	2	0	0
0	0	1	2	−2	0	0	0	0	−2	1	0
0	0	0	0	2	−2	0	0	−1	−2	1	0
0	0	0	0	0	2	−2	−2	2	1	0	0
0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Figure 4.2 The relation between coarse grid image and charge densities

The principal different cases are shown in Figure 4.3. By rearranging the order of the vertices and by multiplying with -1 there will be totally 68 different cases. From this figure the following conclusions can be drawn: 1) An edge crosses a side of a square, if the two end-ponts have different signs. 2) The edge crosses the side closest to that vertex which has the greatest absolute magnitude.

If the image is not bilevel, a suitable charge-grid can be obtained by subtracting a low-pass-filtration of ρ_{cg} from itself. This will give a start value for partion of ρ_{cg} into a) dipole-lines ρ'_{cg} (edges) and b) charge lines together with the part corresponding to the continuous charge distribution ρ''_{cg} (background). The final partition is done, after the edges are detected.

The edge-detection has been carried out as a least-square-minimization of the difference between an actual square and all possible theoretical squares described above. An edge is considered present, if the error is sufficiently small (see Appendix A).

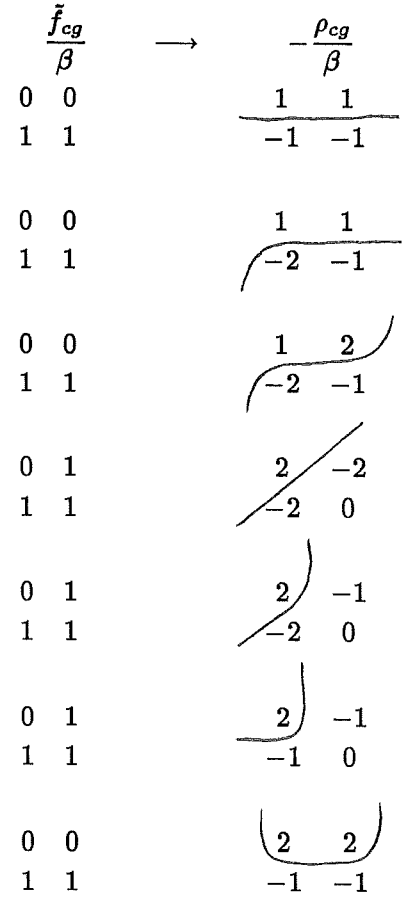

Figure 4.3 Principal different cases of relations between coarse grid image and charge densities. The edges are to be extrapolated in the indicated directions.

4.3 A combined numerical and analytical interpolation scheme

Poisson-solvers have difficulties, when edges are present. Edge-detection partitions ρ_{cg} into ρ'_{cg} (edges) and ρ''_{cg} (background). The Poisson-equation is linear. Rather than solving

$$\triangle \hat{f}_{fg} = -\rho_{fg}$$

numerically, which would be the straight forward approach, it is better to take advantage of the facts mentioned above and solve

$$\triangle \hat{f}'_{fg} = -\rho'_{fg}$$

analytically and

$$\triangle \hat{f}''_{fg} = -\rho''_{fg} \tag{4.1}$$

numerically. This gives $\hat{f}_{fg}$ as

$$\hat{f}_{fg} = \hat{f}'_{fg} + \hat{f}''_{fg}.$$

This is possible, if ρ'_{fg} is supposed to be caused by sharp and straight edges

$\gamma'(q)$ of constant height $\beta(q)$ within each square of the coarse grid. Then

$$\begin{aligned}\hat{f}(x) &= \sum_{q\in\Omega_{cg}} \int_{\gamma'(q)} \beta(q) \frac{\partial K}{\partial n}(x-y)ds_y \\ &= \sum_{q\in\Omega_{cg}} \int_{\gamma'(q)} \frac{\beta(q)}{2\pi} \frac{n(q)\cdot(x-y)}{|x-y|^2} ds_y.\end{aligned}$$

It is easily shown (see Appendix A) that every term in the sum is equal to

$$\pm\frac{\beta(q)}{2\pi}\theta(q;x),$$

where $\theta(q;x)$ is the angle under which the endpoints of $\gamma'(q)$ are seen from the point x. The sign depends on which side of the dipole line x is situated (see Appendix A).

To solve 4.1, the boundary values of $\hat{f}''_{fg}$ are needed. This could be done by linear interpolation of the coarse grid boundary values. To use an iterative method start values are also needed. Both aims can be reached by a bilinear interpolation of the coarse grid. These can be approximately calculated as

$$\hat{f}''_{fg_{start}}(q) = \sum_{p\in\Omega_{cg}} \tilde{f}_{cg}(p)h(p-q) - \hat{f}'_{fg}(q), \quad q\in\Omega_{fg}$$

with h as in 3.1. The last term compensates for the fact that the analytical solution takes no account for boundary values. A possible way to get ρ''_{fg} is to interpolate ρ''_{cg} bilinearly:

$$\rho''_{fg}(q) = \sum_{p\in\Omega_{cg}} \rho_{cg}(p)h(p-q), \quad q\in\Omega_{fg}$$

with h as in 3.1. The application of a conventional Poisson-solver, such as a SOR-routine with Chebychev acceleration (Press et al, 1988, p. 652 ff), could now be done. However this solution needs not attain the initially given values on the coarse grid. To use this extra information, which these vales give, the Poisson-solver has been modified with the constraint

$$\hat{f}''_{fg}(q) = \hat{f}''_{fg_{start}}(q), \quad q\in\Omega_{cg}.$$

This constraint will reduce of the number of equations in the Poisson-solver by the number of points in Ω_{cg}. The result is an approximately polynomial interpolation of

$$\hat{f}''_{cg_{start}}(q) = \hat{f}''_{fg_{start}}(q), \quad q\in\Omega_{cg}$$

of degree 3.

4.4 An analytical interpolation scheme

Another way to obtain an edge-preserving scheme, is to modify a fast scheme, such as the bilinear one. One way to do this is to avoid interpolating over edges once they are detected.

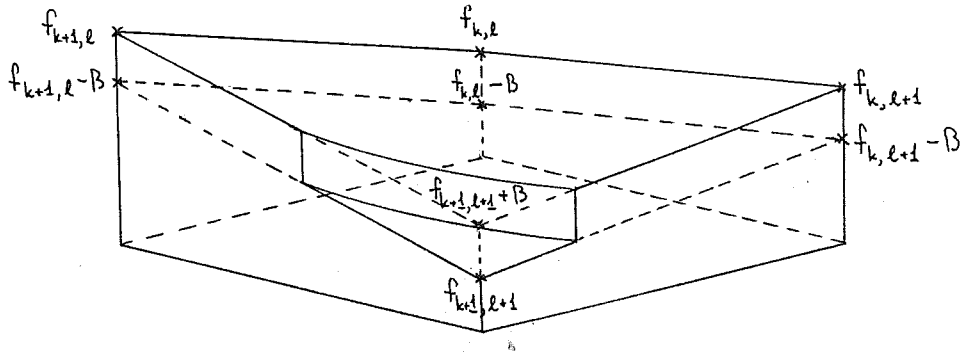

Figure 4.4 How an edge is preserved in the modified bilinear interpolation scheme

Just as in the previous section the knowledge of the edge-height $\beta(q)$ and the edge-cross-points in a square q is used. If an edge is present, the scheme in Section 3.3 can be modified with a test of on which side of the edge x is situated. If x is on the high side, then all f_{ij} on the low side are replaced with $f_{ij}+\beta(q)$, otherwise all f_{ij} on the high side are replaced with $f_{ij}-\beta(q)$. This is described in Figure 4.4.

5. Experiments

The two non-linear interpolation schemes developed in Chapter 4 will be experimentally evaluated here on two different images. Influences of limitations in the ink jet plotter and the reproduction of this report are discussed. Quality as well as computational costs are taken into consideration.

5.1 Images used

One of the images used in the evaluation is an artificially made image with many edges and little background ("test-image"), and the other is a video-image of three faces with few edges and a lot of background ("group-image"). The test-image has specifically been designed to invoke both typical image structures and also worst-case features such as a diagonal line and edges intersecting each other. The diagonal line is a problem for the nonisotropic version of the discrete Laplacian used here.

5.2 Experimental constraints

Due to the limited numbers of intensities in the ink jet plotter, the error-diffusion-technique described in Section 2.2 has been used to avoid level-curves. Unfortunately this tends to blur the result, which can be observed if the result is compared with a video-image. Another constraint is the reproduction of this report, which does not fully justify the ink jet plotter images. Thus the following discussion is not based on the reproduction of the images nor of the original ink jet plotter ones but on video quality ones.

5.3 Evaluation

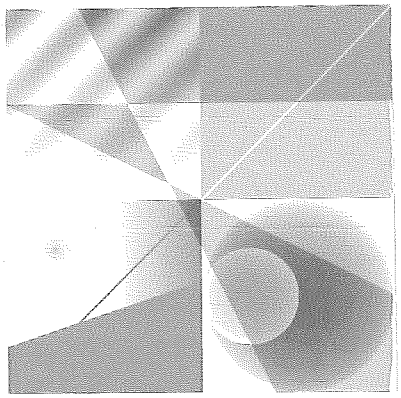

Figure 5.1 Original test-image

For each image the non-linear schemes of Chapter 4 have been compared with a sampled fine grid image, with a pixel replication interpolated one, with a

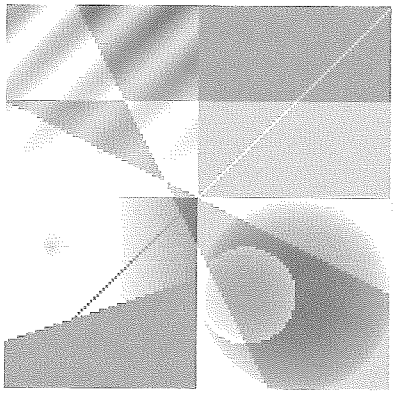

Figure 5.2 Pixel replicated test-image

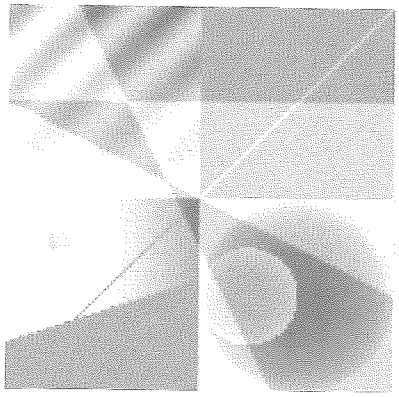

Figure 5.3 Bilinearly interpolated test-image

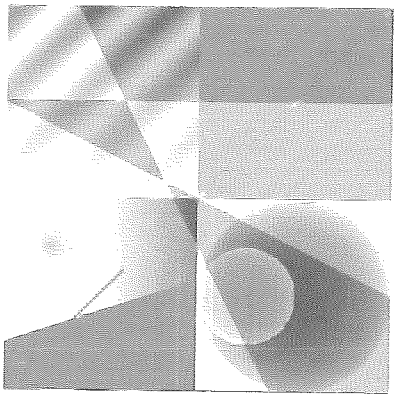

Figure 5.4 Test-image interpolated with the scheme of section 4.3

bilinearly interpolated one and with each other. The approximate technique of deriving a sampled coarse grid image from a fine grid one described in Section 3.2 (with 2σ = the shortest distance between two pixels) has only been used with the group-image, since the other one is not a video-image. The ratio between fine grid and coarse grid has been 4 in both horizontal and vertical directions. The results are presented in the 10 figures beginning with Figure 5.1 .

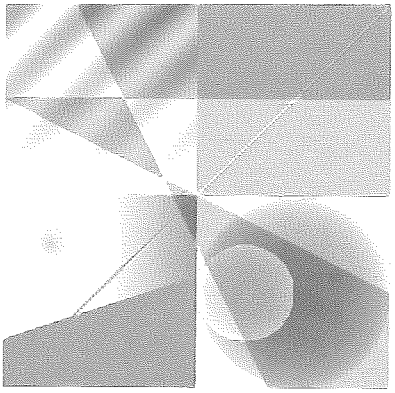

Figure 5.5 Test-image interpolated with the scheme of section 4.4

Figure 5.6 Original group-image

Figure 5.7 Pixel replicated group-image

5.4 Discussion

Test-image

It is easily seen that the non-linear interpolation schemes are superior to the linear ones. Where the height of the edges are not to small, they are well preserved both regarding height and direction, which is seen when they are compared with the fine grid image.

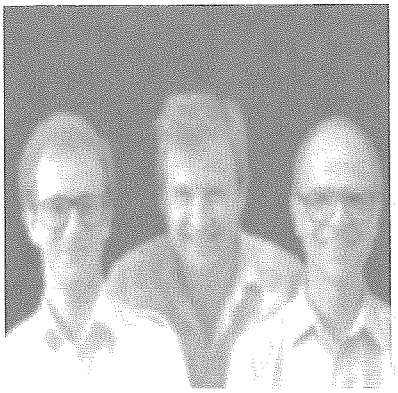

Figure 5.8 Bilinearly interpolated group-image

Figure 5.9 Group-image interpolated with the scheme of section 4.3

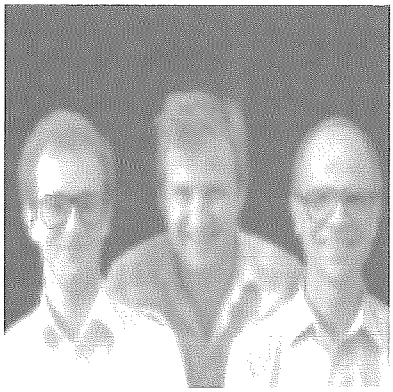

Figure 5.10 Group-image interpolated with the scheme of section 4.4

The line is not treated to satisfaction. However it is the worst case, and if the line had had another direction, it would have been preserved. To fully take care of lines another discretization of the Laplace-operator should have been used. It is also seen that when edges are situated close to each other the edge-detection fails. To overcome this problem more elementary cases than the 68 ones treated should be considered.

The scheme of Section 4.3 seems to add extra edges to the image. This is due

to truncations of the analytic calculations and can be overcome with so called window-functions, which are frequently used in signal-processing. The scheme of Section 4.3 also seems to preserve the direction of the edges less good than that of Section 4.4. This is probably due to the discretization in the fine grid when the angles are computed.

Since the computational costs are less for the scheme of Section 4.4, this scheme must be considered to be the superior one for images of this type.

Group-image

The non-linear interpolation schemes does not seem to preserve edges in a to the eye satisfactory way. The scheme finds extra edges where there are no edges in the original. These extra edges are some what disturbing, and the bilinearly interpolated image is superior but not perfect. The edge-detection seems to be to local, and more sophisticated methods should be used.

Since the computational costs are smaller for the bilinear scheme, this scheme must be considered to be the superior one for images of this type.

6. Conclusions

The ink jet plotter poses interesting problems in how to reproduce images taken by a video camera. A first contribution of the work has been the suggestion to combine a nonlinear transformation from bit code to intensity with an error-diffusion-technique. The technique shows an improvement independent of the interpolation used. The major part of the work has been to investigate a potential-theory-approach to interpolation with the purpose to make edge-preserving interpolation schemes. In particular an explicit description of images by means of a small number of primitives has been obtained. An analytical scheme that maintains most of the positive features with the combined numerical and analytical scheme has also been designed. This scheme has smaller computational cost.

Not all possibilities with the approach have been explored, but the approach and the experiments clearly indicate that improvements are possible and that extensions in diferent directions should be done. It would be interesting to investigate how to preserve edges situated close to each other and particularly lines. This could perhaps be facilitated by studying other discretizations of the Laplace-operator. Another interesting continuation would be to make a more global edge-detection, to get more connected edges.

7. References

ABDOU, I. E. and K.Y. WOUNG (1982): "Analysis of Linear Interpolation Schemes for Bi-Level Image Applications," *IBM J. Res. Develop.*, **26 no. 6**, 667–680.

BERNSTEIN, N. (1976): "Digital image processing of earth observation sensor data," *IBM J. Res. Develop.*, **20**, 40–57.

BJÖRKENGREN, U., C. H. HERTZ and B. SAMUELSSON (1985): "Color Ink Jet Printing—High Resolution and Compound Jet Applications," *Journal of Image Technology*, **11**, 1.

ERIKSSON, T. and L. NIELSEN (1989): "Window based interactive image processing," SSAB Symposium, March 15-16, Göteborg, Sweden.

HANSSON, A., G. SPARR and L. NIELSEN (1989): "Nonlinear interpolation from video images to high quality plotters," SSAB Symposium, March 15-16, Göteborg, Sweden.

KELLOGG, O. D. (1954): *Foundations of Potential Theory*, Dover Publications, Inc., 180 Varick Street, New York, N.Y. 10014 USA.

KERNIGHAN, B. W. (1978): *The C Programming Language*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632 USA.

NIELSEN, L. (1985): "Simplifications in visual servoing," Ph.D. thesis CODEN: LUTFD2/TFRT-1027, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

PRATT, W. K. (1978): *Digital image processing*, John Wiley & Sons, Inc., New York / Chichester / Brisbane / Toronto.

PRESS, W. H., B.P. FLANNERY, S.A. TEUKOLSKY and W.T. VETTERLING (1988): *Numerical Recipies in C—The Art of Scientific Computing*, Press Syndicate of the University of Cambridge, The Pitt Building, Trumpington Street, Cambridge CB2 1RP 32 East 57th Street, New York, NY 10022 USA.

ROSENFELD, A. and C. KAK (1982): *Digital Picture Processing*, Academic Press, Inc., 111 Fifth Avenue, New York, New York 10003, USA.

SPARR, G. (1984): *Kontinuerliga System*, Lund Institute of Technology—Department of Mathematics, Lund, Sweden.

A. Mathematical supplements

In this appendix some of the mathematics computations omitted in the previous chapters are treated.

A.1 Obtaining a coarse grid from a fine one

See section 3.2 for background. One may obtain f_{cg} from $\tilde{f}_{fg}$ as

$$\tilde{f}_{cg}(q) \approx \frac{L_{x_1} L_{x_2}}{2\pi M N R_{x_1} R_{x_2} (\sigma_{cg}^2 - \sigma_{fg}^2)} \sum_{p \in \Omega_{fg}} \tilde{f}_{fg}(p)\, e^{-\frac{(q_1-p_1)^2+(q_2-p_2)^2}{2(\sigma_{cg}^2-\sigma_{fg}^2)}},$$

$$q \in \Omega_{cg}.$$

To see this, let

$$H_{cg}(\omega) = e^{-\frac{(\omega_1^2+\omega_2^2)\sigma_{cg}^2}{2}}, \quad \omega \in \mathbb{R}^2$$

and

$$H_{fg}(\omega) = e^{-\frac{(\omega_1^2+\omega_2^2)\sigma_{fg}^2}{2}}, \quad \omega \in \mathbb{R}^2.$$

Then

$$\tilde{F}_{cg}(\omega) = \tilde{F}_{fg}(\omega)\frac{H_{cg}(\omega)}{H_{fg}(\omega)} = \tilde{F}_{fg}(\omega)\, e^{-\frac{(\omega_1^2+\omega_2^2)(\sigma_{cg}^2-\sigma_{fg}^2)}{2}}$$

$$\Updownarrow$$

$$\tilde{f}_{cg}(x) = \int_{\Omega_c} \tilde{f}_{fg}(y) \frac{1}{2\pi(\sigma_{cg}^2 - \sigma_{fg}^2)} e^{-\frac{(x_1-y_1)^2+(x_2-y_2)^2}{2(\sigma_{cg}^2-\sigma_{fg}^2)}} dy$$

$$\approx \frac{L_{x_1} L_{x_2}}{2\pi M N R_{x_1} R_{x_2} (\sigma_{cg}^2 - \sigma_{fg}^2)} \sum_{q \in \Omega_{fg}} \tilde{f}_{fg}(q)\, e^{-\frac{(x_1-q_1)^2+(x_2-q_2)^2}{2(\sigma_{cg}^2-\sigma_{fg}^2)}}, \quad x \in \Omega.$$

A.2 Poisson-equation with dipole-line charge

See Sections 4.2 and 4.3 for background. Let

$$\rho'_{fg}(x) = -\beta \left[\frac{\partial \delta}{\partial n}\right]_{\gamma'}(x), \quad x \in \Omega$$

be a straight dipole line with density $-\beta$ and moment n. Since the equation is translation and rotation invariant, it sufficies to consider

$$\gamma' : t \to (t, 0), \quad t \in \{s \mid x_{1_1} \leq s \leq x_{1_2}\}$$

and

$$n = (0, -1).$$

Straight forward calculations give

$$\int_{\gamma'} \frac{\beta}{2\pi} \frac{n \cdot (x-y)}{|x-y|^2} ds_y = -\int_{x_{1_1}}^{x_{1_2}} \frac{\beta}{2\pi} \frac{x_2}{(x_1-t)^2 + x_2^2} dt$$

$$= -\frac{\beta}{2\pi} \int_{x_{1_1}}^{x_{1_2}} \frac{\frac{1}{x_2}}{\left(\frac{x_1-t}{x_2}\right)^2 + 1} dt = \frac{\beta}{2\pi} \left[\arctan\left(\frac{x_1-t}{x_2}\right) \right]_{x_{1_1}}^{x_{1_2}}$$

$$= \frac{\beta}{2\pi} \left(\arctan\left(\frac{x_1 - x_{1_2}}{x_2}\right) - \arctan\left(\frac{x_1 - x_{1_1}}{x_2}\right) \right).$$

A.3 Edge-detection

See Section 4.2 for background. Let the true charge densities be stored in the column vector y_0 and the 68 theoretic charge densities with unit height in the collumn vectors $y_i, \quad i = 1, \ldots, 68$. The error ε between the true densities and the densities of a theoretic edge of type i and height β may be expressed as

$$\varepsilon = y_0 - \beta y_i$$

An edge of type i is considered present if y_i is the vector that minimizes

$$\min_{\beta} \left(\varepsilon^T \varepsilon\right) = \min_{\beta} \left((y_0 - \beta y_i)^T (y_0 - \beta y_i)\right)$$

and if the minimal value is sufficiently small. The minimization over β is a least-squares-problem, which has the solution

$$\beta = \frac{y_i^T y_0}{y_i^T y_i}.$$

The minimization over y_i has been carried out as scanning all 68 possibilities.

B. Users manual

The purpose of this manual is only to give a brief glimpse of how to use the Image-interpolation-program. For a more detailed information the reader is referred to the program-listing.

B.1 Getting started

The compiled program is at present stored under the name `image.out`, which is also the name to be typed to run it. SUNVIEW should be installed for the program to work. The memory is limited. If it is too small change the value of `MAXW` and `MAXH` in the file `constants.c` to 2048, recompile the program with `compimage.out` and relink it with `linkimage.out`. This will reduce the allocation of memory to 1/4 for some of the matrixes, but it will also make it impossible to interpolate images greater than 256×256 pixels. No interpolated images will be greater than 2048×2048 pixels. If the constants mentioned above are not altered, the greatest possible image-size to interpolate is 512×512 pixels and the greatest possible image-size is 2048×2048 pixels.

The criterion of a successful start is the appearence of a panel with 18 buttons. To chose an item in the panel, click with the left mouse button. It is possible to partition the items into 5 groups corresponding to:

- Conversion, dispaying and listing of images
- Elementary operations
- Interpolation schemes
- Listing and deletion
- Exit

Before proceeding to the different items an explanation of file-types, and file-headers would be helpful. The following types exist:

`byte`: the image representation for video images—1 byte per pixel

`float`: the image representation on which computations in this program are made—4 bytes per pixel

`ascii`: the image representation, which makes printing of numbers in an image possible—5 ASCII characters = 5 bytes per pixel

`ink`: the image representation for the ink jet plotter—2 bytes per pixel

`edge`: the representation for detected edges in an image—20 bytes per pixel

`imp`: the representation for impulse responses—4 bytes per pixel

`bit`: internal

`short` internal

Files of type `short` should not exist after termination of the program. Do not delete files of type `float` outside the program without deleting the eventually corresponding `bit`-file. Some of the file types have file headers—`float`, `edge`, `imp`, `bit` and `short`. The headers consists of 2 integers describing width and height of the file and for files of type `imp` there are 2 more integers describing

the lengths of the impulse response in the coarse grid. File-names and image-names may consist of any combination of characters, as long as they are not longer than 58 characters, and as long as the operating system does not complain. No warnings will be given, if longer names are used. Unpredictable errors may occur. The program will internally use the different file-types as file-extensions, which makes it possible to use the same name for different files as long as they are of different types.

B.2 Conversion, displaying and listing of images

Extract image

If say every fourth pixel in an image are to be skipped or if a part of an image is to be taken out, it can be done with this item. After answering questions about from which images of type byte or float to extract from and to save to and how many pixels to skip in the x (horizontal) and y (vertical) directions, the image to extract from is displayed. Positioning the arrow at the desired upper left corner and clicking with the left mouse button, positioning at the lower right corner and clicking with the middle one and when satisfied clicking with the right one will cause the extraction to begin.

Display image

By selecting this item and typing the name of the image of byte or float type to display, after a while a bilevel image will occur on the screen.

List numbers in image

This item makes it possible to list numbers in an image of byte or float type on the screen. After answering the question of what image to list from, just do as in Extract image when the image becomes visible.

Convert image to ASCII file

Since most printers only read ASCII-characters, this is the item to use, when to print intensity levels of an image. It works just as the previous item, but instead of maing a listing on the screen, it will produce a listing on a file of type `ascii`. Do not forget to specify what name to save under !

Convert image to video representation

To view images on a video-screen, use this item to convert an image of type byte or float to type byte. After answering questions about from which image to read from and which one to save to, select error-diffusion if desired and what number of quantization levels to use. This feature makes it possible to simulate an ink-image on a video-screen by using error-diffusion and only 16 quantization levels instead of 256 and explains why it is possible to convert from type byte to byte.

Plot images

As many as 6 images of type float or byte may be written to the same plot. There is no check that there will be enough space. The images are written from left to right and from up to down in the order they are named. The available plot space is 3900×2800 pixels. If there was not enough space, no information will be given about what images were not written. First specify how many images to write and then the names of the images and if error-diffusion is desired or not. Finally specify what name to save the plot under.

List impulse response file

By answering the question of the desired name, information about length, magnification and actual values will be given.

B.3 Elementary operations

Math-utility

Here all elementary mathematical operations and functions are implemented. Three different types of operations exist:

matrix op matrix: $result(i,j) = m_1(i,j)\,\mathrm{op}\,m_2(i,j)$

matrix op scalar: $result(i,j) = m_1(i,j)\,\mathrm{op}\,scalar$

f(matrix): $result(i,j) = f\,(m(i,j))$

After selecting what type of operation wanted, questions are asked about name(s)—eventually values—of operands and name to save the result under. For the first two types possible operations are: $+$, $-$, $\cdot$ and $/$. For the third type possible operations are: asinh, acosh, atanh, exp, log, log10, fabs, sin, cos, tan, asin, acos, atan, sinh, cosh, tanh, sqrt and cbrt.

Convexity-estimates

This item partitions an image into regions where the Hessian of the image is positive definite, negative definite, positive semidefinite, negative semidefinite and indefinite pixels. This is done by producing five subimages side by side in a big image in the order above. A black pixel in the most left subimage means that that pixel is positive definite, a black pixel in the subimage right to the most left one means negative definite and so on. The only questions to answer are where to read from and where to save to.

Rotate image

To rotate an image $\pi/2$ use this item ! Just answer the questions about where to read from and where to save to.

B.4 Interpolation schemes

Interpolate image

In this item analytical interpolation schemes are implemented such as

1. Convolution
2. Bilinear
3. Bilinear - Edge
4. Edge

Answer the questions of where to read from and where to save to. In the 3 first cases the coarse grid image is where to read from and in the last case the edge-image is meant. In all cases the fine grid image is where to save to.

1. Convolution There are three different ways to get an impulse response to convolve with:

1.1. Use predefined procedures

1.2. Read from file

1.3. Type on keyboard and write to file

In 1.2 just specify the name of the file under which the impulse response is saved—these are created in 1.3. In 1.1 and 1.3 specify the length of the window-function and then what type of window to use. Available types are

1. Rectangular window
2. Hamming window
3. Hanning window
4. Blackman window

Then specify magnification. In 1.1

1. $\text{sinc}(x)\,\text{sinc}(y)$
2. $HP(alfa + 0.05)$
3. Zero order hold

may be selected while in 1.3

1. Read whole matrix $h(i, j)$ form keyboard
2. Read $h(middley, j)$ and $h(i, middlex)$ from keyboard
3. Read $h(middle)$ from keyboard

may be selected.

2. Bilinear This is a faster implementation of the bilinear interpolation scheme than the one that can be implemented by means of a convolution. Here just specify magnifications.

3. Bilinear - Edge This is an implementation of the edge preserving bilinear interpolation scheme. Specify the name of the edge-image and magnifications wanted.

4. Edge This is an implementation of the solution to $\triangle f = \rho$ where ρ are dipole lines. Specify magnifications and windowlengths.

Edge-detection

In this item edge-detection is done. Specify names for the Laplace-image to detect edges for and names to save the edge-image and the new Laplace-image under.

Solve Poisson-equation

In this item the Poisson-equation described in section 4.3 is solved. Enter name of Laplace- and startvalue-image, name of where to save to and magnifications.

B.5 Listing and deletion

List names of files

After entering file-type, all files of that type are listed.

List file-header

After entering file-type and file-name the header is listed.

Delete file

After entering file-type and file-name the file is deleted. If type is `float` then the eventually corresponding bit-file is also deleted.

B.6 Exit

This item terminates the program.