Modeling the Television Process

Michael Anthony Isnardi

Technical Report 515

May 1986

Massachusetts Institute of Technology
Research Laboratory of Electronics
Cambridge, Massachusetts 02139

Modeling the Television Process

Michael Anthony Isnardi

Technical Report 515

May 1986

Massachusetts Institute of Technology
Research Laboratory of Electronics
Cambridge, Massachusetts 02139

This work has been supported in part by Members of the Center for Advanced Television Studies, an industry group consisting of the American Broadcasting Company, Ampex Corporation, Columbia Broadcasting Systems, Harris Corporation, Home Box Office, Public Broadcasting Service, National Broadcasting Company, RCA Corporation, Tektronix, and 3M Company.

Modeling the Television Process

by

Michael Anthony Isnardi *

Submitted to the Department of
Electrical Engineering and Computer Science
in May, 1986 in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

Abstract

Advances in digital signal processing have led to a renewed interest in basic television research. Much of this attention is devoted to the processing of the video signal in the channel, while past research has focused on analytical models of particular television devices. In this thesis, a general spatiotemporal model is developed that includes all important camera, channel, and display processes. Numerical simulation is used to study the effect of system parameters on image quality at the display. TVSIM, a TV simulation program, is developed as a tool for the design and analysis of a general monochrome baseband TV system. It models (1) the image illuminance falling on a camera target, (2) the physical scanning operation of a tube-type or solid-state camera, (3) the generation, processing, and transmission of the video signal, and (4) the physical scanning operation of a tube-type or solid-state display. The final output is an image representing the detailed luminance pattern on a hypothetical display. This image may be further processed by a model of the human visual system, or it may be compared with other processed images.

TVSIM and other simulation software are used to study a number of basic television processes. Several charge scanning models are developed and compared; it is shown that destructive readout in camera tubes enhances high-frequency detail while introducing phase distortion. The effect of camera and display apertures on image quality is investigated. Simulation results show how physically different camera lag processes can yield similar lag responses. Temporal integration patterns are used to explain and model the motion rendition characteristics of common imaging devices. Standard, extended-definition, and high-definition TV systems are simulated and compared. Finally, two adaptive interpolation methods for television are described. One method uses Gaussian beam shaping to reduce scan line visibility; the other uses digital contour interpolation to reduce aliasing.

Thesis Supervisor: William F. Schreiber, Professor of Electrical Engineering

---

* AT&T Bell Laboratories Scholar

Acknowledgments

I wish to extend my appreciation to the following: Prof. William F. Schreiber, for his technical insight and supervision of this research; Profs. Campbell Searle, Arun Netravali, and Eric Dubois for their helpful comments on various drafts; AT&T Bell Laboratories, for its financial support of my Ph.D. work; John Ward and Kathy Cairns, for their administrative assistance and Monday morning smiles; John Wang, Jean-Pierre Schott, and Steve Corteselli, for maintaining the computers; Steve Berczuk, Gordon Shaw, and Haw-minh Lu, for their diligent research assistance; Anne Kirkland, for her editorial suggestions; and all my friends at CIPG and DSPG, for the many kindnesses that made my stay a pleasant one.

Finally, I wish to thank my dearest friends for their understanding and encouragement throughout my academic years, and to my family for their love and support.

Table of Contents

Title Page 1
Abstract 2
Acknowledgments 3
Table of Contents 4
List of Figures 6
Notation 11

1. Introduction 14
1.1. The Need for a General Model of the Television Process 15
1.2. Description of a Complete Model of Television 16
1.3. Review of Previous Research 18
1.4. Objectives of this Research 20
1.5. Organization of the Thesis 21

2. A Spatial Model of a Monochrome Baseband TV System 23
2.0. Introduction 23
2.1. Functional Modeling vs. Circuit Modeling 23
2.2. A Spatial Model of a Television System 24
2.3. The Camera 24
2.3.1. The Lens 26
2.3.2. Light-to-Charge Conversion 29
2.3.3. Charge Readout 31
2.4. Gamma Correction 46
2.5. The Channel 47
2.6. The Display 47
2.6.1. Display Gamma 48
2.6.2. Charge-to-Light Conversion 48
2.7. The Human Visual System 49
2.8. Experiments 50
2.8.1. Simulation of Tube-Type TV Systems 51
2.8.2. Effect of Apertures in Solid-State TV Systems 59
2.8.3. Simulation of the HVS Perceptual Response 76
2.9. Summary 80

3. A Temporal Model of a Monochrome Baseband TV System 82
3.0. Introduction 82
3.1. A Temporal Model of a Television System 82
3.1.1. Modeling the 1-D Image Illuminance Function 84

3.1.2. Modeling the Dynamics of a Camera Element 84
3.1.3. Modeling the Dynamics of a Display Element 90
3.1.4. Temporal Models of the Human Visual System 92
3.1.5. Experiments 94
3.2. A 3-D Spatiotemporal Model of a Television System 109
3.2.1. 3-D Temporal Integration Patterns 109
3.2.2. Modeling Charge Spreading 118
3.2.3. Experiments 119
3.3. Summary 126

4. Adaptive Image Interpolation for Television 127
4.0. Introduction 127
4.1. Analog Gaussian Beam Shaping 128
4.1.1. Linear Shift-Variant 2-D Filtering 129
4.1.2. Normalization of Filtered Sequences 132
4.1.3. Adaptive Filter Parameters 133
4.1.4. Finding Edges in Images 135
4.1.5. Experimental Results 139
4.2. Adaptive Digital Line Interpolation 158
4.2.1. Nonadaptive Methods 158
4.2.2. Motion-adaptive Method 161
4.2.3. Novel Block-Matching Method 161
4.3. Summary 169

5. TVSIM - A TV Simulation Program 172
5.0. Introduction 172
5.1. Operation of TVSIM 172
5.1.1. Modes of Operation 173
5.1.2. TVSIM Components 175
5.2. TVSIM Experiments 182
5.2.1. 64-line Simulations 184
5.2.2. High-Resolution Simulations 184
5.3. Summary 192

6. Conclusions and Suggestions for Further Research 193

References 197
Appendix A - Documentation for TV Simulation Programs 204
Appendix B - UNIX Datfiles 227
Appendix C - Modulation Depth of Gaussian Beam 229

List of Figures

1.1. A block diagram of a generalized television system.

1.2. A block diagram of a general monochrome baseband television system.

2.1. Identifiable spatial processes in a monochrome baseband television system.

2.2. Image formation by a camera lens.

2.3. Lens simulation. (a) Object plane image. (b) Focused focal plane image, with aberration. (c) Defocused focal plane image, no aberration. (d) Defocused focal plane image, with aberration.

2.4. Electron beam scanning geometry.

2.5. Simulation of three charge erasure models. (a)-(d) Snapshots of Gaussian beam linearly erasing positive charge image. Normalized video signals derived using (e) passive filtering, (f) linear erasure, and (g) exponential erasure.

2.6. Simulation of the beam sharpening effect. (a) Original charge image. (b) Passive filtering. (c) Linear erasure. (d) Exponential erasure.

2.7. Perspective snapshots of linear charge erasure. (a) Gaussian electron beam about to enter wall of positive charge. (b)-(d) Remaining charge at three different beam positions.

2.8. Effect of beam width and amplitude on rise and lead time for constant charge amplitude ($A=1$). (a) Passive filtering. (b), (d) Linear erasure. (c), (e) Exponential erasure.

2.9. Effect of charge amplitude on rise and lead time for constant beam parameters ($B_0=B_1=1$,$\sigma=3$). (a), (c) Linear erasure. (b), (d) Exponential erasure.

2.10. Effect of beam amplitude on image detail using linear erasure. B_0 = (a) 25, (b) 50, (c) 75, and (d) 100 charge units.

2.11. Effect of beam amplitude on image detail using a vidicon tube. Beam current is low in (a) and is nearly normal in (b).

2.12. General structure of solid-state array.

2.13. (a) HVS spatial frequency response. (b) HVS spatial impulse response.

2.14. Effect of Gaussian beam widths on image quality. Natural image.

2.15. Effect of Gaussian beam widths on image quality. Test pattern.

2.16(a). CMAN original (256H×256V).

2.16(b). Read beam: impulse; write beam: impulse

2.16(c). Read beam: impulse; write beam: 4H×4V prism

2.16(d). Read beam: impulse; write beam: 1H×7V triangle

2.16(e). Read beam: impulse; write beam: 7H×7H pyramid

2.16(f). Read beam: 1H×23V $\frac{\sin x}{x}$; write beam: 1H×23V $\frac{\sin x}{x}$

2.16(g). Read beam: impulse; write beam: 9H×9V Gaussian

2.16(h). Read beam: impulse; write beam: 11H×11V Gaussian

2.16(i). Read beam: impulse; write beam: 13H×13V Gaussian

2.16(j). Read beam: impulse; write beam: 15H×15V Gaussian

2.16(k). Read beam: impulse; write beam: 7H×15V Gaussian

2.16(l). Read beam: impulse; write beam: 3H×15V Gaussian

2.16(m). Read beam: 3H×3V Gaussian; write beam: 3H×15V Gaussian

2.16(n). Read beam: 3H×9V Gaussian; write beam: 3H×15V Gaussian

2.16(o). Read beam: 3H×15V Gaussian; write beam: 3H×15V Gaussian

2.16(p). Read beam: 9H×9V Gaussian; write beam: 3H×15V Gaussian

2.17. Effect of square aperture sizes on image quality. Natural image.

2.18. Effect of square aperture sizes on image quality. Test pattern.

2.19. Perception of spatial detail using linear bandpass HVS model. (a) Original image. (b) Perception of (a) at a distance where height of image subtends $\approx 2°$ of arc. (c) Image displayed using 64 scan lines. (d) Perception of (c) at same viewing distance.

3.1. Identifiable temporal processes in a single camera and display element.

3.2. (a) Temporal integration pattern. (b) Temporal interpolation pattern.

3.3. Typical phosphor decay curve.

3.4. (a) HVS temporal frequency response. (b) HVS temporal impulse response.

3.5(a). Linear TV system with no lag and with $T_I = T_F$.

3.5(b). Linear TV system with no lag and with $T_I = 0.25 \cdot T_F$.

3.5(c). Linear TV system with photoconductive lag.

3.5(d). Linear TV system with beam discharge lag and 90% readout efficiency.

3.5(e). Linear TV system with beam discharge lag and 50% readout efficiency.

3.5(f). Vidicon-type TV system with photoconductive lag and beam discharge lag.

3.5(g). Plumbicon-type TV system with photoconductive lag and beam discharge lag.

3.5(h). Linear TV system with beam discharge lag and 90% readout efficiency.

3.5(i). Lag-free TV system with $\gamma_1 = 0.65$ and $\gamma_D = 2.2$.

3.5(j). Signal-dependent lag without bias light.

3.5(k). Improvement of signal-dependent lag with 10 units of bias light.

3.6. Stability of charge dynamics under different readout mechanisms. (a) Linear readout of 4500 charge units. (b) Exponential readout with 90% efficiency.

3.7. Flicker perception at 30, 60, and 90 Hz. (a) Display output. (b) Linear HVS response. (c) Log HVS response.

3.8(a). Movie film or progressive frame-transfer CCD.

3.8(b). Progressive tube-type video camera.

3.8(c). Progressive line-transfer CCD.

3.8(d). Interlaced line-transfer CCD.

3.8(e). Interlaced interline-transfer CCD.

3.8(f). Interlaced frame-transfer CCD.

3.9. Display snapshots of a 3-D TV simulation.

3.10. Motion rendition of moving disk. $T_I = T_F$.

3.11. Motion rendition of moving disk. $T_I = 0.25 \cdot T_F$.

3.12. Motion rendition of revolving disks. $T_I = T_F$.

3.13. Motion rendition of oscillating disk. $T_I = T_F$.

4.1. Discrete model of the television process.

4.2. Linear shift-variant FIR filters (ROS size: 5×5).

4.3. 2-D Gaussian filter parameters.

4.4. A 2-D surface with normal vector and tangent plane.

4.5. Subset of a 2-D Gaussian filter matrix.

4.6. Circle test pattern.

4.7. Isotropically filtered versions of Fig. 4.6(d).

4.8. Edge information derived from Fig. 4.6(b).

4.9. Adaptively filtered versions of Fig. 4.6(d) using various edge strength metrics.

4.10. Adaptively filtered versions of noisy original using various edge strength metrics.

4.11. Cameraman test image.

4.12. Isotropically filtered versions of Fig. 4.11(d).

4.13. Edge information derived from Fig. 4.11(b).

4.14. Adaptively filtered versions of Fig. 4.11(d) using edge metrics derived in various ways.

4.15. Effect of beam eccentricity on image quality when edge orientation is known exactly.

4.16. Effect of beam eccentricity on image quality when edge orientation is known exactly.

4.17. Effect of normalization on image quality when edge orientation is (a) known exactly, (b) derived from original, (c) derived from subsampled image, then linearly interpolated, and (d) derived from linearly interpolated subsampled image.

4.18. Simulation of adaptive line interpolation. (a) 64 simulated scan lines, displayed on a standard CRT. (b) 4:1 adaptive line interpolation of video signal used in (a), displayed on a high-resolution CRT.

4.19. Line interpolation for 2:1 interlaced signals.

4.20. Spatial sampling geometry for 2:1 line interpolation.

4.21. 2:1 line interpolation from odd field, natural image. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.

4.22. Spectra of images in Fig. 4.21. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.

4.23. 2:1 line interpolation from odd field, test pattern. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.

4.24. Rms error as a function of block size and range of search.

4.25. Performance of block-matching algorithm at various levels of subpixel accuracy. (a) Original (64×64). (b) Odd field of (a). (c) 2:1 line replication. (d) 2:1 line averaging. Adaptive block matching ($B=7$, $R=1$) at accuracy levels of (e) whole pixel, (f) half pixel, (g) one-third pixel, (h) one-quarter pixel.

5.1. Block diagram of TVSIM program.

5.2. TVSIM modes of operation.

5.3. (a) Effect of p/σ ratio on modulation of flat field and single blanked line. (b) Luminance values used to determine modulation depth. (c) Modulation depth curves for flat field (*MF*) and single blanked line (*ML*).

5.4. Simulation of four combinations of cameras and displays. (a) Tube-type to tube-type. (b) Solid-state to tube-type. (c) Tube-type to solid-state. (d) Solid-state to solid-state.

5.5. Effect of number of scan lines on image quality. (Left half) NTSC quality. (Right half) HDTV quality.

5.6. Effect of line interpolation on image quality. (a) Original. (b) NTSC quality. (c) 2:1 line-interpolated NTSC. (d) HDTV quality.

5.7. Effect of line interpolation and sharpening on image quality. (a) NTSC quality. (b) 2:1 line-interpolated NTSC with sharpening. (c) HDTV quality. (d) 2:1 line-interpolated HDTV with sharpening.

5.8. Improved image quality using wobble raster. (Left half) wobble off. (Right half) wobble on.

Notation

A	beam amplitude scaling factor
B	block size in adaptive block-matching algorithm
B_0	beam amplitude in linear erasure model
B_1	beam amplitude in exponential erasure model (A/m)
G	photoconductive gain
γ_1	photoconductive gamma of camera target
γ_2	external gamma correction in camera
γ_D	display gamma
η_R	readout efficiency of electron beam in exponential model
h_0	scaling factor for temporal impulse response of display
H_A, V_A	horiz. and vert. size of active area of resolution cell (m)
H_C, V_C	horiz. and vert. size of entire resolution cell (m)
θ	edge or beam orientation (rad)
i,j,k,l,m,n	discrete spatial or temporal variables
I	discrete interpolation factor
i_d	dark current density (A/m^2)
k_L	scale factor for lag equation (sec^{-1})
L, S	long- and short-axis half-lengths of Gaussian beam
M	magnification of optical system
M_1, M_2	size of ROS of discrete FIR filters
N	frame number
$\mathbf{N}$	surface normal
N_H, N_V	number horiz. and vert. cells in solid-state sensor
q_B	amount of charge deposited by beam during readout (C/m^2)
q_d	time-integrated dark current density (C/m^2)
R	range parameter in adaptive block-matching algorithm
RC	RC time constant of photoconductor (sec/m^2)
ρ, ω	digital frequency variables (rad)
σ_H, σ_V	horiz. and vert. widths of 2-D Gaussian beam
σ_r, σ_w	widths of circularly-symmetric Gaussian read and write beam
t, τ	continuous temporal variables (sec)
τ_D	decay time constant of display phosphor (sec)

τ_L	photoconductive lag time constant (sec)
T_D	display interval (sec)
T_I	integration period (sec)
T_F	frame period (sec)
T_O	time offset for pixel integration (sec)
T_R	readout interval (sec)
T_S	storage interval (sec)
ϕ	normal angle
x, ξ	continuous horizontal spatial variables (m)
y, η	continuous vertical spatial variables (m)
x_0, y_0	location in xy-plane
$b_C(x,y,t)$	perceived brightness function of HVS in focal plane of camera
$b_D(x,y,t)$	perceived brightness function of HVS at display screen
$\beta(x,y)$	edge strength metric
$B(x,y;\xi,\eta)$	beam current density when beam is centered at (ξ,η) (A/m^2)
$B'(x,y)$	beam current density (A/m^2)
$e_C(x,y,t)$	camera focal plane illuminance (lux)
$f(\cdot,\cdot)$	general 2-D function
$h(x,y)$	general spatial impulse response function
$h_D(t)$	temporal impulse response of display
$h_{HVS}(x,y,t)$	spatiotemporal impulse response of HVS
$\theta(x,y)$	edge orientation metric
$i_G(t)$	input current to display (A)
$i_L(t)$	lagged photocurrent density (A/m^2)
$i_P(t)$	photocurrent density (A/m^2)
$i_R(x,y)$	output current density (A/m^2)
$i_R(t)$	output current from camera (A)
$l_O(x',y',t)$	object plane luminance (cd/m^2)
$l_D(x,y,t)$	display luminance (cd/m^2)
$L[\cdot]$	linear filter operation at display
$q_I(x,y)$	time-integrated positive charge density on photoconductor surface (C/m^2)
$q_R(t)$	leftover charge density (C/m^2)
$q_I(x,y;\xi,\eta)$	positive charge density on photoconductor surface when beam is centered at (ξ,η) (C/m^2)

$q'_I(x,y)$	positive charge density on photoconductor surface (C/m^2)
$T'[\cdot]$	filter operation at camera
$T[\cdot]$	filter operation at display
$u(n_1,n_2)$	discretized image in camera focal plane
$v(t)$	video signal into channel (V)
$v'(t)$	video signal from channel (V)
$V(x,y,t)$	voltage distribution with respect to cathode on camera target (V)
$x(n_1,n_2)$	discretized zero-padded image at display
$y(n_1,n_2)$	discretized interpolated image at display

Chapter 1

Introduction

The limited spatiotemporal response of the human visual system (HVS) makes television feasible. Under certain constraints, the HVS can be made to perceive a continuous natural scene when in fact it is looking at a luminance pattern discontinuous in both time and space. Unfortunately, present TV transmission standards, which have not changed in over thirty years, do not fully satisfy these constraints. In some cases, too much information is sent, and in other cases, too little. For instance, the NTSC[1] system exhibits a gross mismatch between horizontal and vertical color detail [1], and the PAL[2] system has annoying large-area flicker [2].

Much has been learned since the early days of television. Models for the HVS have been proposed and tested, camera and display technology has progressed steadily, and digital signal processing (DSP) has found widespread use. With this knowledge and experience, TV engineers are reconsidering the basic problem of designing a broadcast TV system that simultaneously maximizes image quality and minimizes transmission bandwidth.

Recently, sophisticated signal processing in the receiver has resulted in enhanced-definition TV (EDTV), that is, the improvement of picture quality within the present broadcast standards. For example, it is now possible to greatly enhance the displayed image by spatiotemporal interpolation [3], adaptive noise removal [4], and better color separation [5]. A more radical proposal is known as high-definition TV (HDTV). To provide improved picture quality at the transmitter, HDTV may require incompatible broadcast standards. The Japan Broadcasting System (NHK) has already demonstrated an HDTV system that displays more finely resolved imagery at the expense of increased channel bandwidth [6]. It is not clear that the NHK or any other proposed HDTV standards offer the best tradeoff between image quality and bandwidth.

---

[1] National Television Systems Committee, the North American TV standard.

[2] Phase Alternation by Line, the main European TV standard.

A question fundamental to modern television design is: given the channel bandwidth, the signal-to-noise ratio (SNR), and the viewing conditions, what are the "best" camera, channel, and display standards? Much theoretical work has focused on this problem, but reducing theory to practice is difficult, and only a few broadcast TV standards have emerged. Clearly, the search for an optimal system would be aided by having some method of adjusting TV parameters and quickly reviewing the results.

1.1. The Need for a General Model of the Television Process

A general software model of the television process is a desirable tool for TV design and analysis. Such a model would accept an arbitrary set of TV system parameters and would produce, among other data, an output image representing the luminance pattern on a hypothetical display. Ideally, a general model would accurately simulate every photoelectric or electronic process in every system component and would run in real time. This, of course, is too much to hope for at present. A more modest goal, which is the subject of this thesis, is an off-line simulation of fundamental processes in a monochrome[3] baseband TV system.

Television image quality is difficult to define and measure. In the early years of television, research focused on the required line and frame rates for acceptable response at the display [7]. In the 1940's, after transmission standards had been set, attention was devoted to the system parameters affecting picture quality [8, 9]. Some authors developed analytical models to explain the behavior of cameras and displays [10]. Others devised models to study the theoretical effect of a few important system parameters on image quality [11]. Very often, the device models were made linear, time-invariant, and deterministic for the sake of tractability. Although useful results were obtained, it was evident that more complex models were necessary.

Only recently has technology permitted the numerical simulation of physical processes. Computer modeling finds use in many areas, including lens design and

[3] In this report, *monochrome* is synonymous with *black-and-white*.

video system analysis [12, 13]. A computer simulation helps to refine an analytical model of a system or a device. Often, a simulation reveals new insight and provides a deeper understanding of a complex process. Besides providing solutions to non-linear systems, numerical methods can also be used to evaluate model accuracy and performance. The effect of system parameters on image quality can be measured, and parameters can be made independent, which may be difficult to achieve in practice. Another advantage of computer simulation over purely analytical methods is that natural imagery can be used as data, and the processed images can be compared to the output of real devices.

The digital processing of images and image sequences requires a large amount of computation. Fortunately, high-speed processors with megabytes of main memory are becoming affordable. The decrease in simulation run-time, coupled with the ease of changing system parameters, makes software simulation a viable alternative to hardware prototypes. Furthermore, prototypes that offered such flexibility would be prohibitively expensive, as would the building of many small, dedicated hardware systems for optimization and comparison.

1.2. Description of a Complete Model of Television

Fig. 1.1 depicts a general high-level model of the television process. A possible software implementation for each component is discussed below.

Scene. Ideally, a scene model would be an animated, three-dimensional (3-D) color representation of natural objects in a natural environment. Sophisticated computer graphics techniques such as ray tracing are needed to render realistic scenes. Because these scene models have essentially infinite bandwidth, the filtering and geometrical transformations of optical components can be studied. The scene model need not be this complicated, however. Simplifications include eliminating color, motion, or depth; if all three are eliminated, one can model a flat, static, monochromatic scene such as a resolution test chart, or a black-and-white telecine.

Optical System. The optical system transforms the infinite bandwidth scene into a finite bandwidth planar image. If a complex 3-D scene is being rendered, a

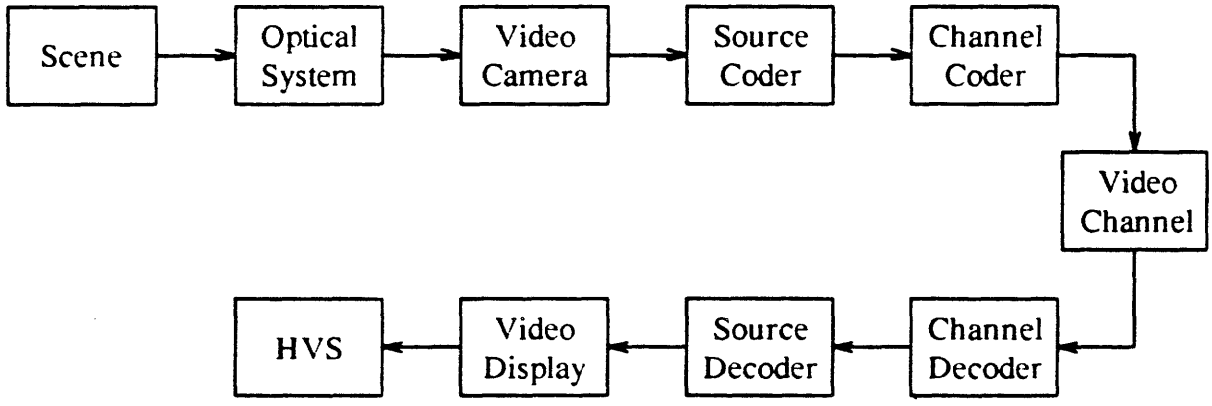


Fig. 1.1. A block diagram of a generalized television system.

nonideal lens model is most naturally incorporated into the rendering process itself. This takes care of depth-of-focus phenomena, as well as geometric and chromatic aberrations. However, for a flat, monochromatic scene, the lens model may be computed as a separate processing step: it simply filters the two-dimensional (2-D) object plane image, producing a scaled and possibly blurred focal plane image.

Video Camera. The video camera transforms the imaged illuminance into a one-dimensional (1-D) video signal. Newer solid-state cameras are slowly replacing the older tube-type units, and accurate models do not yet exist for either of these. Basic to the operation of any photoelectronic imaging device is light-to-charge conversion; a good model for this complex process is necessary. For tube-type cameras, one needs to model electron beam erasure; for solid-state devices, charge transfer. A simplified camera model would include only these important processes. A more sophisticated model would incorporate secondary effects such as beam bending, geometric distortion, transfer inefficiency, and noise sources.

Source Coder. The source coder preconditions the video signal for transmission. Gamma correction, sharpening, color encoding, and digitization are performed at this stage. Statistical or visual encoding may also be applied.

Channel Coder. The channel coder attempts to compensate for known channel degradations. It equalizes and modulates the video signal for transmission. If the channel is digital, error correction coding may be performed.

Video Channel. The channel conveys the signal from transmitter to receiver. It may support a digital bit stream or an analog waveform. A good model would include bandwidth and noise characteristics, as well as other common degradations, such as fading and multipath reflection (ghosting).

Channel Decoder. The channel decoder complements the conditioning performed by the channel coder. It demodulates the signal and attempts to remove channel degradations.

Source Decoder. The source decoder complements the conditioning performed by the source coder. It may compensate for display or observer characteristics.

Video Display. The video display transforms the 1-D video signal into a 2-D luminance pattern that is continuously refreshed and updated. Displays may be tube-type (CRT's) or solid-state (LCD's), monochrome or color. The essential operations of scanning and charge-to-light conversion must be accurately modeled.

Human Visual System. A spatiotemporal model of the HVS completes the generalized television simulation. Although it is difficult to predict the perceived quality of a single image, it is easier to predict the relative quality between a processed image and its original, or between two processed images derived from the same original. A simple model would ignore the complex interactions between spatial and temporal pathways in the human visual system. For many applications, a linear bandpass filter in both the spatial and temporal dimensions is sufficient to predict relative image quality.

1.3. Review of Previous Research

The analysis of television systems is not new. In the early days of television, Mertz and Gray [14] developed a mathematical theory of scanning using certain simplifying assumptions. They were able to derive the shape of the video signal spectrum, as well as some of its important properties. Schade [1] and Huck, *et. al.*

[15] investigated the effect of spot intensity profiles and photosensor aperture shapes on both the blurring and aliasing of images. Much of their work is formulated in the frequency domain. More recently, Potmesil and Chakravarty [16] extended the pin-hole projection geometry to model the effect of a lens and an aperture function of an actual film camera on the rendering of synthetic imagery. They did not simulate the TV scanning process; however, their results affirmed the feasibility of numerically simulating camera lenses and apertures.

Destructive readout in image tubes was first investigated by Miller and Izatt [17]. This is one of the few attempts in the literature to model the aperture function as a non-passive, destructive process. They develop a simple linearized model of the readout process that incorporates progressive erasure of the stored image. The self-sharpening effect of the beam was studied in detail, but only simple analytic models were used. More recently, Selke [10] and Kurashige [18] extended the analysis to include other beam and camera characteristics.

Some important work has been done in analyzing the physical properties of photoconductors used as image targets in camera tubes. For instance, Schade, in one of his many classical studies of television, analyzed resolution properties and quantum processes in TV cameras [19, 20], and the electron optics and signal readout of return-beam vidicons [11]. Van de Polder [21] studied target stabilization effects in television pick-up tubes and derived certain requirements for the capacitance and voltage swing of plumbicon targets to obtain an image showing no discharge lag. Nudelman [22] did a theoretical analysis of noise in vidicon and orthicon image tubes that accounts for factors such as storage, frequency response, resolution and efficiency.

The modeling of video displays and the perception of displayed information has also received some attention. An optical simulation of scanning line disturbance was carried out by Wolcott [23] and Engstrom [24]. The subjective effect of scan line visibility was observed, but the simulation was only approximate in that it modeled a system with impulse-like sampling for both camera and display beams. Pica's [25] model of a color display incorporated as parameters pixel size, amplifier response, gamma, spot profile, and duty cycle of line-phosphor stripes. Schade

[26] studied image reproduction by a line raster process.

Modeling the human visual system [27] has received attention recently. Carlson and Cohen [28] developed the "window of visibility" model of the HVS for predicting what an observer would perceive when viewing a monochrome display. Klopfenstein and Carlson [29] investigated the perceptual properties of color monitors.

There have been several attempts to numerically simulate complete television systems. Of significance is the work done by Perlman at RCA [30], who modeled a linearized NTSC television system from camera to display. Modulation transfer functions (MTF's) of the camera lens, read beam, and write beam were applied to the signal in the frequency domain. Because of computer memory limitations, only two horizontal TV lines were generated and processed, so the whole simulation is limited to vertically equivalent TV signals such as color bars and other test patterns. More recently, Palmer and Shenoy [31] at COMSAT simulated TV transmission of NTSC and MAC[4] signals over a satellite channel. Once again, only two lines of video test signals were considered. Wong [32] developed a comprehensive computer simulation of the generation, transmission and reception of composite color NTSC signals.

1.4. Objectives of this Research

There are two major goals of the present research. The first is the development of a general model of monochrome baseband television, including all important processes in the camera, channel, display, and HVS. The second is the application of the model to the design, analysis, and optimization of proposed TV systems. Fig. 1.2 is a simplified block diagram of the television system considered in this thesis. Although noise is present in all real electronic devices, it is only considered in the channel model. In a typical TV system, channel noise is the most significant image degradation.

[4] Multiplexed Analog Components, a method of time-multiplexing the color components of a video signal to eliminate cross-color and cross-luminance effects.

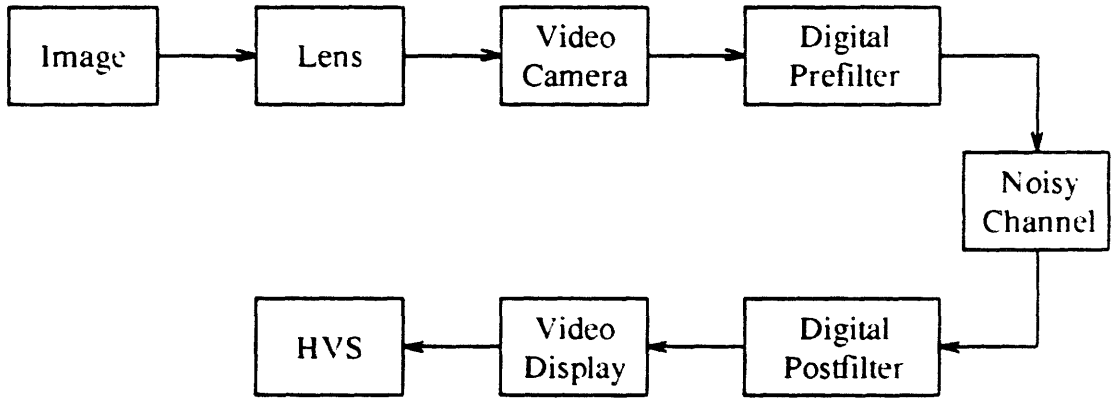


Fig. 1.2. A block diagram of a general monochrome baseband television system.

The model is restricted to a *monochrome baseband* TV system because the extension to color is relatively straightforward, and because many degradations associated with RF modulation can be simulated in baseband. Image coding is not considered, but it can be inserted easily into the baseband model. Unlike previous efforts, this simulation accounts for nonlinearities in the camera and the display, incorporates common degradations, and processes *entire images* as well as standard video test signals. Also, the model allows one to "zoom in" and examine the fine-grain structure of apertures and scanning lines in cameras and displays.

1.5. Organization of the Thesis

The major portion of this thesis describes the television process in a mathematical framework that is amenable to modeling on a digital computer. Chapter 2 examines the spatial properties of video cameras and displays. The effect of aperture shapes on perceived image quality is examined, and the effect of destructive readout in camera tubes is investigated. Chapter 3 treats similar issues in the temporal domain. Temporal integration patterns are introduced and are used to explain the differences in motion rendition among various imaging devices. Chapter 4

describes how adaptive image interpolation can be applied at the receiver to minimize aliasing and line visibility. Chapter 5 describes the operation of TVSIM, a TV simulation program that models the spatial properties of electronic cameras and displays. NTSC, EDTV, and HDTV systems are simulated and compared in terms of displayed image quality. In Chapter 6, the main results are summarized and followed by suggestions for further research.

Chapter 2

A Spatial Model of a Monochrome Baseband TV System

2.0. Introduction

This chapter models the spatial characteristics of a monochrome baseband TV system. The spatial and temporal properties cannot be completely separated, since spatial variations are converted into temporal variations by the scanning process. For example, the temporal frequency response of the channel affects spatial reproduction at the display. In practice, spatial properties are measured by televising a *static* scene, usually a resolution test chart. This chapter deals exclusively with static scenes illuminated by unchanging monochromatic light. Chapter 3 will incorporate temporal processes into the model.

2.1. Functional Modeling vs. Circuit Modeling

Many of the camera and display models reported in the literature start with an equivalent *circuit model* of a specific imaging device. For instance, a small area of a photoconductive target is often modeled by a capacitor in parallel with a light-dependent resistor [33, 34]. Some Plumbicon models replace the resistor with a light-dependent current source [35]. The electron beam in camera tubes has been modeled by a grounded commutating switch [35], a nonlinear resistor [36], or a current source, depending on the application. Although it enables accurate simulation of internal processes, a circuit model is quite device-specific: equivalent circuits for different devices may differ in both element values and circuit topology. For this reason, comparative analysis is difficult. However, waveforms associated with different devices are remarkably similar in *shape,* even though the underlying physical processes may be quite different. These waveforms can often be modeled

by mathematical functions whose parameters are device-specific. To develop a general TV simulator, it is necessary to abandon circuit models and instead work with generalized *functional models*. The key to general television modeling is to identify common photoelectric processes that are accurately characterized by a small number of parameters.

2.2. A Spatial Model of a Television System

To begin, it is important to identify those processes in cameras and displays that affect the spatial resolution of the perceived image. Ignoring noise, these processes are shown in the block diagram of Fig. 2.1. The *lens* spatially scales the object plane image and may blur off-axis points. In camera tubes, the *light-to-charge conversion* in the photoconductive target, besides being nonlinear, may result in a charge image of lower spatial detail. The *read beam* in a camera tube, and the *resolution cell* in a solid-state sensor, filter and sample the focal plane illuminance; spatial aliasing may occur if the camera aperture is too small relative to image detail. The combined *frequency response* of video amplifiers and channel tend to degrade horizontal response. The *write beam* in a display tube and the *resolution cell* in a solid-state display both filter and interpolate the video signal; each may impose a visible sampling structure on the reproduced image. Finally, the spatial characteristics of the *human visual system* produce a perceptual response that depends on image content and viewing conditions. Because it is still poorly understood, the HVS is difficult to model, but it must be considered for completeness. Mathematical models for each process are described below.

2.3. The Camera

A video camera converts a focused optical image into an equivalent charge image that is integrated over time and scanned, producing a video signal. As explained in Chapter 3, the phase relationship between temporal charge integration and readout for each picture element (pixel) is device dependent. For static scenes, however, the phase difference is insignificant and will not be considered here. For

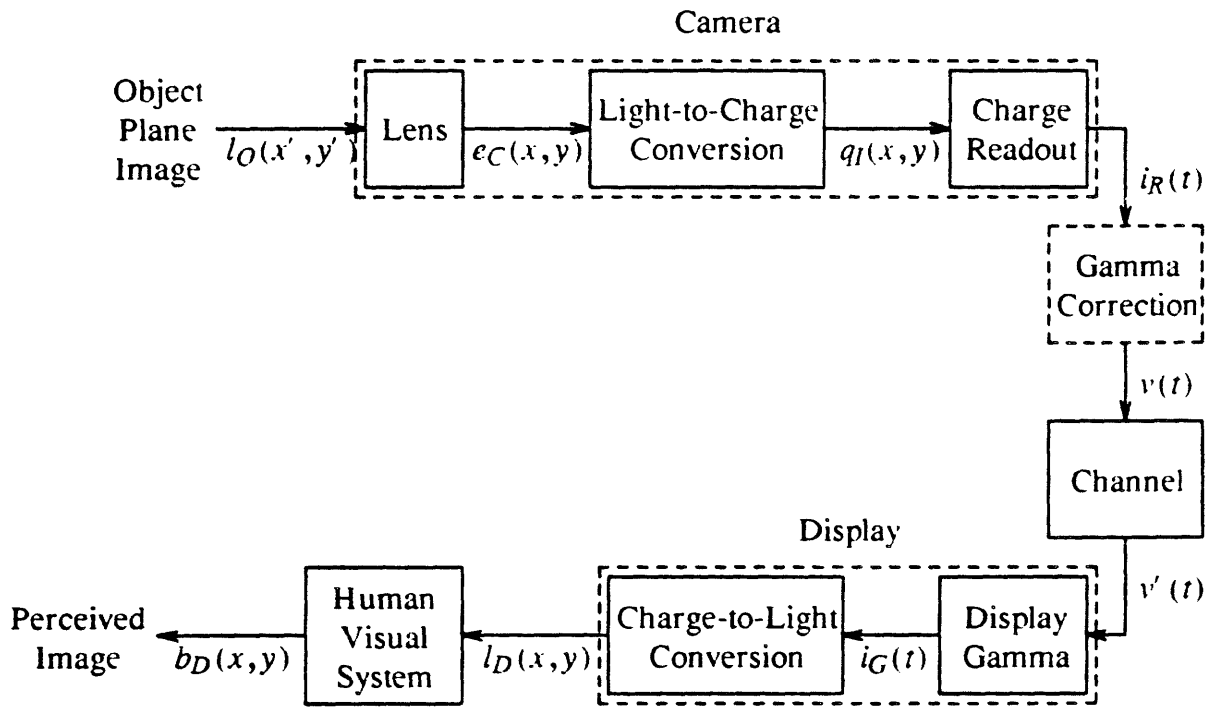


Fig. 2.1. Identifiable spatial processes in a monochrome baseband television system.

simplicity, this chapter will consider image sensors that operate in *storage-erase* mode [37], which is described as follows:

(1) A shutter opens at $t = 0$.

(2) A static focal plane image is converted into a static charge image.

(3) The shutter closes at $t = T_I$, where T_I is the *integration period.*

(4) The charge image is filtered and sampled, and each sample is read out one at a time in a predetermined order.

The model assumes a static scene, 100% charge readout, no charge spreading [38], no beam bending [39], and no noise, implying that all video frames are identical. Thus only one video frame is needed for analysis. For many high-quality

imaging devices, such as Plumbicon tubes and charge-coupled devices (CCD's), these assumptions are valid. However, for noisy, laggy devices, such as vidicon tubes, several frames of video must be averaged in order to produce a representative video frame.

2.3.1. The Lens

The lens focuses an image of the scene on the camera's photosensitive target. Conceptually, the action of the lens may be viewed as a two-step process as shown in Fig. 2.2. First, the *scene luminance*,[5] related to the *scene illuminance*[6] as well as the *reflectance* of objects in the scene, is projected onto an object plane some distance in front of the lens. The z-dependence is suppressed by this projection. The static monochromatic luminance distribution in the object plane is denoted by $l_O(x', y')$. This function is then transformed by the lens into an illuminance distribution $e_C(x,y)$ in the image, or focal, plane. The spatial variables (x', y') denote coordinates in the object plane; (x,y), the image plane. To avoid the unnecessary complications caused by image inversion, the object and image planes are placed on the same side of the lens as shown in Fig. 2.2. This simple recomposition of geometry corresponds to a viewpoint behind the image plane, so that the image occurs right side up [40].

Under incoherent light, optical systems are linear in intensity [41]. The object luminance and image illuminance distributions for an *ideal*, linear, spatially invariant incoherent imaging system are related by [42]

$$e_C(x, y) = \frac{1}{M^2}\, l_O\left(\frac{x}{M}, \frac{y}{M}\right) . \qquad (2.1)$$

This equation states that the focal plane image predicted by geometrical optics is a spatially scaled replica of the object plane image. Here M, a dimensionless quantity, is the *magnification* of the system: $(x, y) = (Mx', My')$. The factor $1/M^2$ scales

[5] Commonly used units of luminance are the *lumen per steradian per square meter* and the *footlambert*.

[6] Commonly used units of illuminance are the *lux* (lumen per square meter) and the *footcandle* (lumen per square foot).

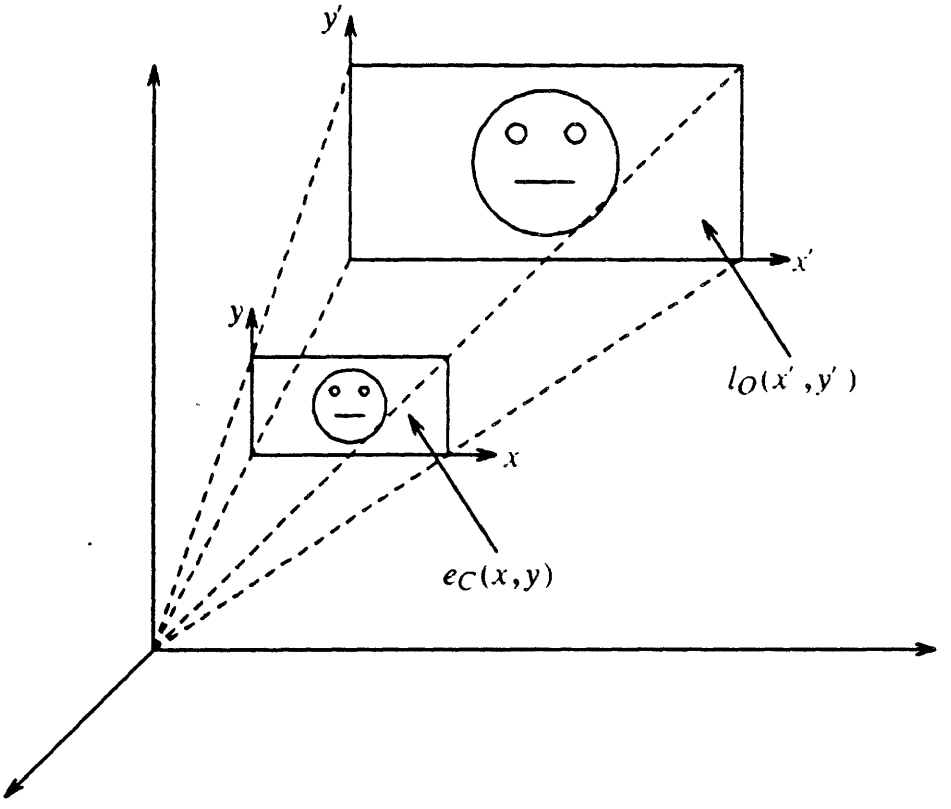


Fig. 2.2. Image formation by a camera lens.

the brightness.

Physical lenses blur the image as well as scale it. A linear spatially *invariant* imaging system is described by a convolution integral:

$$e_C(x,y) = \int_{x'}\int_{y} |h(x-x',y-y')|^2 \, l_O(Mx',My')\,dx'\,dy' \;, \qquad (2.2)$$

where $h(x,y)$ is the Fourier transform of the pupil function [42], which characterizes the lens shape. For instance, a circular thin lens has a pillbox shaped pupil function; for a particular wavelength, the square of its Fourier transform

magnitude is the well known Airy pattern [43, 44]. When Airy patterns are integrated over all wavelengths in the visible range, a function approximating a Gaussian is produced [45]. The squared magnitude appears in the expression because optical intensities, rather than amplitudes, are involved. $|h(x,y)|^2$ is also known as the lens *point spread function* (PSF).

Typically, focal sharpness degrades as one moves off-axis [41]. The system is then described by a linear *spatially variant* PSF. The imaging relationship becomes

$$e_C(x,y) = \int_{x'}\int_{y'} |h(x,y;x',y')|^2 \, l_O(Mx', My') \, dx' \, dy' \quad . \tag{2.3}$$

As mentioned above, a lens may both *scale* and *blur* the object plane image. For spatially continuous signals, magnification is nothing more than a scaling of the spatial variables. For spatially discrete signals, this requires sampling rate conversion [46]. Even if a lens is not considered in the camera model, the digitized object plane image will usually undergo a sampling rate conversion in order to map its points onto the resolution grid of the camera target. Accordingly, any blurring caused by the camera lens can most easily be applied at this stage. From signal processing theory, it is known that the "ideal" interpolation filter for an image sampled on a Cartesian grid is a 2-D $\sin(x)/x$ function; however, for image reconstruction, this may produce negative values in the focal plane illuminance distribution, which is impossible with a real lens. In practice, other interpolation filters, such as the sharpened Gaussian [47], produce better results. The blurring operation of the lens can easily be incorporated into the sampling rate conversion process by convolving the lens PSF with the interpolation filter. This produces an overall lens PSF that scales and blurs the image simultaneously.

Figs. 2.3(a) and (c) depict the action of a spatially invariant lens on a grid of dots (luminous impulses). Moderate and severe blurring are shown and are simulated by convolving the original image with a 2-D circularly symmetric Gaussian function.

All lenses exhibit aberrations of some kind. Five types of monochromatic aberrations have been identified: *spherical aberration, coma, astigmatism, field curvature,* and *distortion* [44]. The first three blur the image, and are less disturbing than the latter two, which deform the image.

A common lens degradation is the imaging of off-axis points as elongated blobs pointing towards the optical axis. Figs. 2.3(b) and (d) are simulations of such a blurring defect. The PSF's of off-axis points are Gaussian ellipsoids whose long axes are radially aligned and whose eccentricities increase with distance from the center. Moderate and severe blurring are shown. The energy within each imaged dot is constant because a lens transmits, but does not absorb, optical energy. This simulation requires linear spatially variant 2-D signal processing; the theory and implementation are discussed in Chapter 4.

2.3.2. Light-to-Charge Conversion

Photoelectric conversion occurs inside a photosensitive planar target, or *mosaic,* consisting of microscopic particles of special compounds that transform incident light into a net positive charge. The term "mosaic" is descriptive of the early camera targets that consisted of a fine wire mesh, lightly sprayed with an insulating material, and filled with photosensitive "plugs" [48]. This term is also an accurate physical description of modern solid-state arrays such as CCD's, since the entire target is a matrix of photosensors. Each photosensor converts incident light to charge and integrates this charge between readout intervals.

The matrix structure of solid-state sensors lends itself easily to discretization and numerical simulation. However, a general model must describe *any* camera target in terms of grid elements because it must eventually be discretized for numerical simulation. In most camera tubes, the target is a continuous sheet of photoconductor that is capacitively coupled to a transparent conductive signal plate [49]. In this case, a *grid element* or *mosaic element* can be thought of as a very small area of the photoconductor target over which the stored charge pattern is essentially constant. The electron scanning beam may cover and discharge many of these mosaic

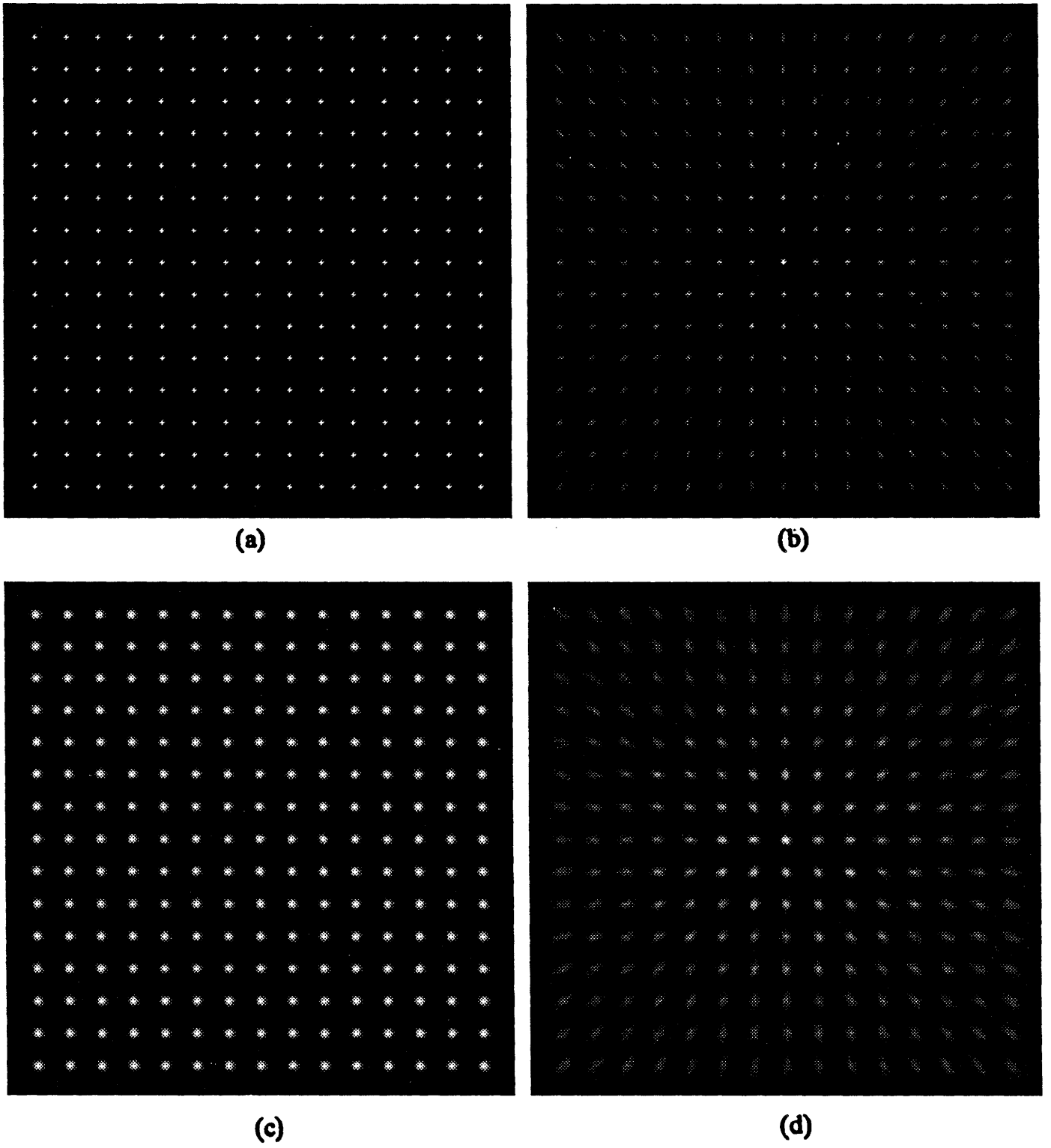


Fig. 2.3. Lens simulation. (a) Object plane image. (b) Focused focal plane image, with aberration. (c) Defocused focal plane image, no aberration. (d) Defocused focal plane image, with aberration.

elements at once [50].

The transformation from light to charge is a complex photoelectric process that has been studied extensively [51, 52, 53, 54]. Some camera targets, such as those found in Plumbicons and silicon-array vidicons, have a nearly linear light-to-charge transfer characteristic [55]. Most targets, however, exhibit a transfer nonlinearity that can be modeled quite well by a power law characteristic [56]:

$$q_I(x, y) = G \cdot h_{ph}(x,y) * e_C(x, y)^{\gamma_1} + q_d \ , \tag{2.4}$$

where $q_I(x,y)$ is the positive charge density formed by temporally integrating the photocharge over the duration of the integration period, G is the *photoconductive gain*, $h_{ph}(x,y)$ is the PSF of the photoconductor [54], $e_C(x, y)$ is the time-integrated illuminance, γ_1 is the *photoconductive gamma*, and q_d is the charge density arising from the time-integrated *dark current*. If $\gamma_1 = 1$ and $q_d = 0$, the device becomes linear.

2.3.3. Charge Readout

Once the charge image is formed, it must be integrated and read out. The physical readout process is different for camera tubes and solid-state devices. In camera tubes, a high-impedance electron beam scans the positive charge pattern, neutralizing the charge image at each point; the discharge current is capacitively coupled to the amplifier circuitry via a transparent conducting signal plate. In solid-state devices, charge is physically transferred from its generation site to an output amplifier by a low-impedance electronic scanning circuit [57].

2.3.3.1. Tube-Type Cameras

Modeling the scanning process as a passive filtering operation is common [56] because it leads to useful results and is amenable to Fourier analysis. However, it is a poor approximation to the physical process of charge erasure. In this section,

the passive filtering model is compared with two erasure models for tube-type scanning.

A mathematical treatment begins by defining a Cartesian coordinate system (x, y) on the camera target with its origin at the center. This is shown in Fig. 2.4. Let $q_I(x,y;\xi,\eta)$ represent the stored positive charge density. The spatial variables ξ and η are needed to reflect the dependency of charge distribution on beam position. Then $q_I(x,y;\xi,\eta)dxdy$ is the charge on an elemental area $dxdy$ when the electron beam is centered at (ξ,η). Denote the current density in the readout beam by the deterministic function $B(x,y;\xi,\eta)$. A more accurate model would include shot noise [38, 58] in the read beam; however, this would only complicate the analysis and is not considered here. Let the beam density be normalized so that the center of the beam has unit amplitude: $B(\xi,\eta;\xi,\eta)=1$. Assume the coordinate axes have been aligned so that η is constant along a scan line. Furthermore, let the beam

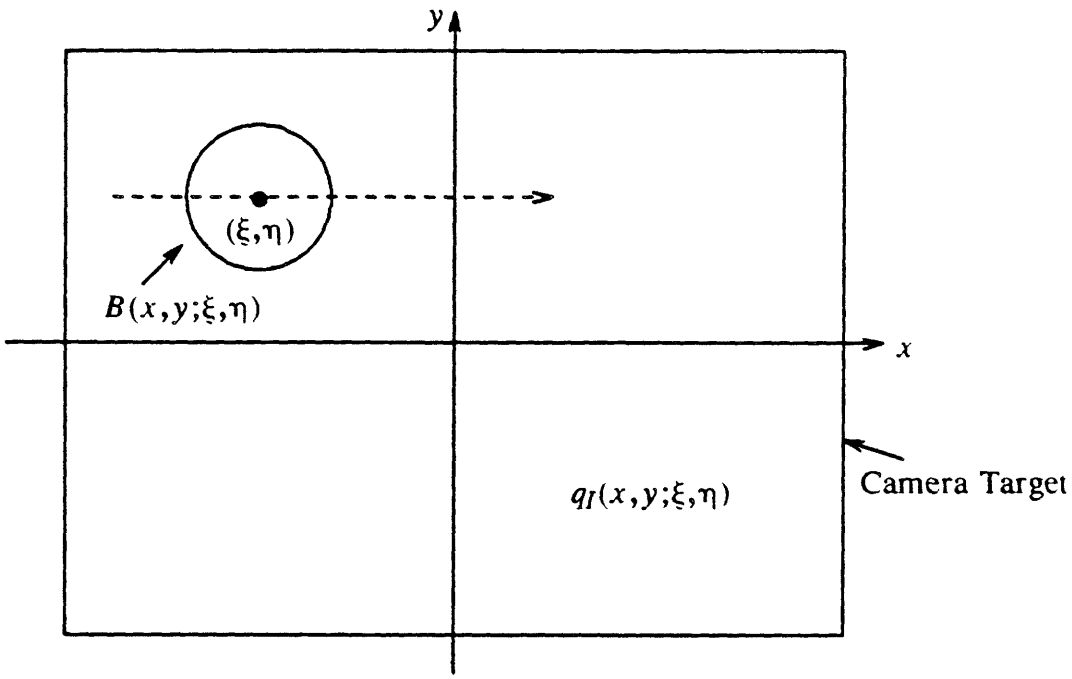


Fig. 2.4. Electron beam scanning geometry.

scan with constant velocity in the ξ direction; this produces a linear relationship between ξ and time.

2.3.3.1.1. Charge Scanning Models

Passive Linear Filtering Model. When modeled as a linear space-invariant filter, the charge density is independent of the position of the beam. The output current density when the beam is centered at location (ξ,η) is given by the convolution of the beam-independent charge density, $q'_I(x,y)$, and the beam current density, $B'(x,y)$:

$$i'_R(\xi,\eta) = B'(\xi,\eta) * q'_I(\xi,\eta) = \iint_{xy} B'(\xi-x,\eta-y)\cdot q'_I(x,y)\,dx\,dy \ . \qquad (2.5)$$

This model is inherently inaccurate because the beam has no effect on the charge density. For instance, if the beam amplitude is scaled, this model predicts an amplitude scaling of the output current density at all frequencies. In reality, changing the beam current does not affect the output level if B is high enough. Furthermore, experiments show that changing the beam current affects the frequency response of the output signal as well. Obviously, a better model is needed.

Linear Erasure Model. The linear erasure model assumes that positive charges on the target are instantaneously neutralized by electrons deposited by the read beam. An example will make this clear. Suppose that there are Q positive charge units at location (x,y) just before scanning, and that B negative charge units are deposited by the beam at that location during scanning. If $Q > B$, then B charge units are removed and $Q - B$ charge units remain at (x,y); however, if $Q < B$, then Q charge units are removed and zero remain. The excess beam electrons are turned back and collected by the anode [59]. This second condition, $Q < B$, is desired because it results in a readout signal that is proportional to the stored charge at each location. This type of readout implies that the local rate of charge reduction is linearly proportional to the beam density alone:

$$\frac{\partial}{\partial \xi} q_I(x,y;\xi,\eta) = -B_0 \cdot B(x,y;\xi,\eta) , \qquad (2.6)$$

where B_0 is the beam amplitude. This equation states that the charge density on the target at "time" $\xi + d\xi$ is equal to the charge density at ξ, minus the charge deposited by the beam at ξ. It is valid for $q_I > 0$. The model is called linear for the following reason. If a beam of uniform current density were to pass at constant velocity over location (x,y), the amount of charge remaining at that location would *linearly* decrease towards zero as the beam moved across it.

Exponential Erasure Model. Miller and Izatt [60] modeled the photoconductive target as a distributed RC network. If Q units of positive charge are stored at location (x,y) just before scanning, then at each instant of time, only a fraction of the remaining charge is removed at that location. The instantaneous beam density at (x,y) determines the fraction removed, and the remaining charge approaches zero but does not become negative. This type of readout implies that the local rate of charge reduction is proportional to the product of the beam and charge densities:

$$\frac{\partial}{\partial \xi} q_I(x,y;\xi,\eta) = -B_1 \cdot B(x,y;\xi,\eta) \cdot q_I(x,y;\xi,\eta) , \qquad (2.7)$$

where B_1 is the beam amplitude (having different units from B_0). If $0 < B_1 < 1$, this equation states that the charge density on the target at $\xi + d\xi$ is equal to a spatially weighted fraction of the charge density at ξ. If a beam of uniform current density were to pass at constant velocity over location (x,y), the amount of charge remaining at that location would *exponentially* decrease towards zero as the beam moved across it.

2.3.3.1.2. Formation of the Output Signal

In both the linear and exponential erasure models, the instantaneous current density is integrated over the beam area to produce the total current:

$$i'_R(\xi,\eta) = \iint_{xy} \frac{\partial}{\partial \xi} q_I(x,y;\xi,\eta)\,dx\,dy \ . \qquad (2.8)$$

Conceptually, the 1-D output signal is formed by sampling the 2-D current density continuously in ξ but discretely in η:

$$i_R(t) = i'_R(H_S \cdot (\mathrm{frac}(t/T_H) - \tfrac{1}{2}),\ V_S \cdot (\mathrm{frac}(t/T_{V'}) - \tfrac{1}{2})) \ , \qquad (2.9)$$

where $H_S \times V_S$ is the size of the target, frac(x) is the fractional portion of the real number x, T_H is the horizontal line period, and $T_{V'}$ is the vertical frame period. Interlace and video blanking are not considered in this formula.

2.3.3.1.3. Experiments

In this section, computer simulations are performed to assess the effects of these three readout models on the frequency response, rise time, and lead time of the video signal.

Frequency Response: Fig. 2.5 is a simulation of three scanning modes. The charge image covers 256× 19 mosaic elements, and is shown along with the beam function in Fig. 2.5(a). The beam is a 2-D Gaussian with $\sigma = 3$ mosaic elements (too large to render much detail!), truncated to size 19× 19 elements. The center of the beam carries 100 negative charge units per element per unit time, and each mosaic element contains between 23 and 255 positive charge units. Figs. 2.5(b), (c), and (d) depict the partially erased charge image when the center of the beam is at a horizontal distance of 24, 120, and 212 mosaic elements. These positions are marked *A*, *B*, and *C* in the graphs below. Figs. 2.5(e), (f), and (g) represent one line of a normalized video signal, each obtained by a different scanning model. Fig. 2.5(e) is the result of a passive linear filtering operation between the beam function and the charge image. Figs. 2.5(f) and (g) show linear and exponential erasure. The smoothest output is produced by passive filtering; the most jagged, by linear erasure. The implications are clear: linear erasure is more responsive to high spatial detail.

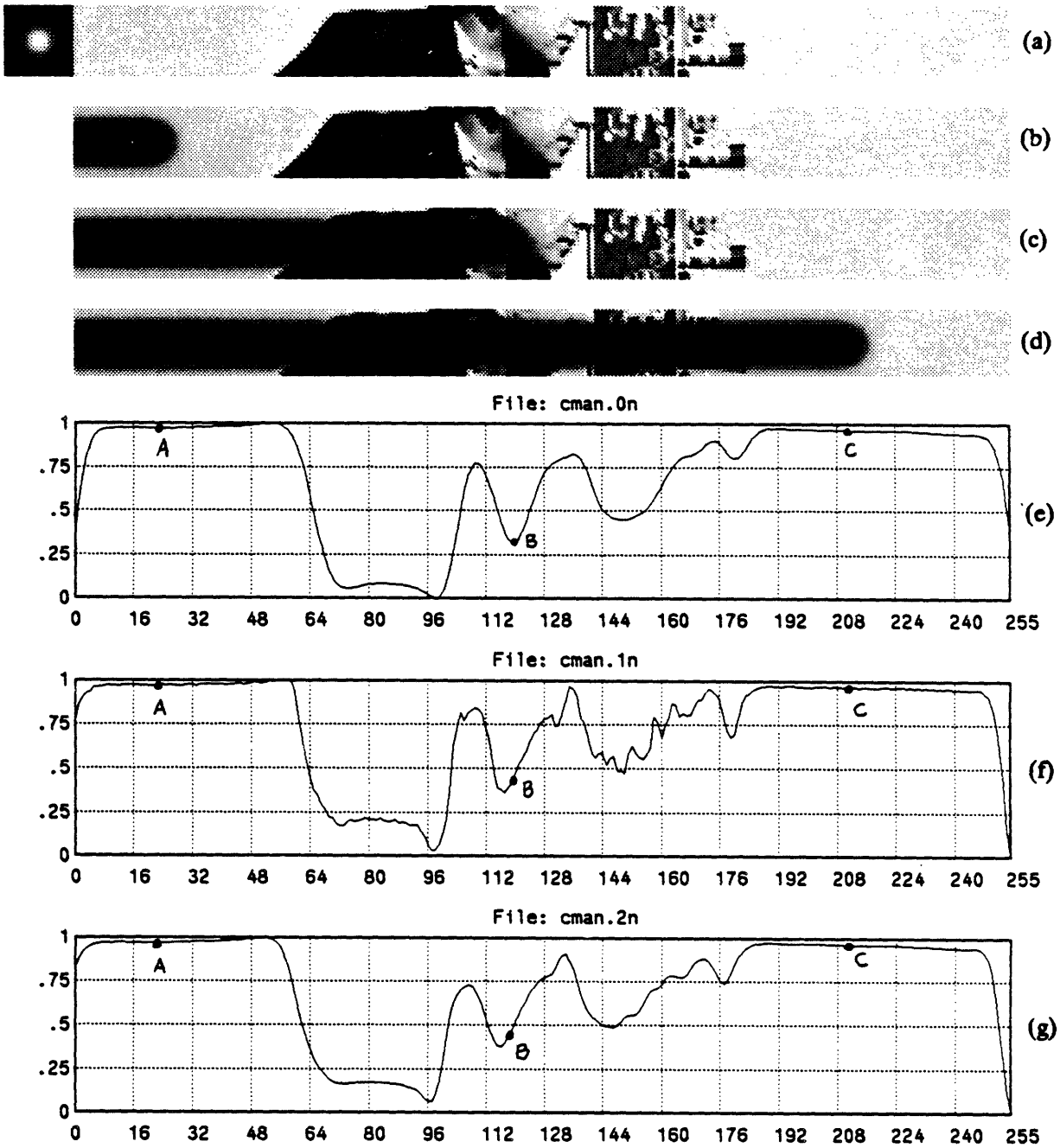


Fig. 2.5. Simulation of three charge erasure models. (a)-(d) Snapshots of Gaussian beam linearly erasing positive charge image. Normalized video signals derived using (e) passive filtering, (f) linear erasure, and (g) exponential erasure.

The Beam Sharpening Effect: The increased high frequency response of destructive readout is called the "self-sharpening" or "beam sharpening" effect [54, 61, 62], further illustrated in Fig. 2.6. All parameters are identical to those in Fig. 2.5, except that the charge image is now a horizontal frequency sweep signal. A horizontal slice through this signal is shown in Fig. 2.6(a). Figs. 2.6(b), (c), and (d) show the results of passive filtering, linear erasure, and exponential erasure. The envelope of the decaying sweep is proportional to the Fourier transform magnitude of the processed signal [41]. Note the extended response of the linear and exponential erasure models. This illustrates the beam sharpening effect of charge erasure. The linear erasure model, while more sensitive to high spatial detail, also causes higher distortion.

The sharpening can be explained as follows. As the beam moves over and erases the stored charge pattern, its leading edge neutralizes most of the charge; therefore, the effective aperture size is much smaller than the actual area. The smaller effective aperture yields a higher frequency response, but its asymmetry introduces some phase distortion. Recently, Kurashige [63] performed a detailed analysis of the self-sharpening effect that includes beam and target characteristics as well as scanning standards.

Rise Time and Lead Time: There are three important parameters in erasure modeling when using a circular Gaussian spot: the spot width (σ), the amplitude of the beam current (B_0 or B_1), and the amplitude of the charge image (A). The following experiment illustrates the effect of beam width and amplitude on rise and lead time when the amplitude of the charge pattern is held constant. Consider a circular Gaussian spot sweeping across a time-integrated charge density described by a step function in the x-direction. Fig. 2.7 shows perspective "snapshots" of the remaining charge with the beam at four different positions along an x-oriented scan line. Linear charge erasure is being performed.

Now consider the output current produced by scanning a vertical strip of charge. The effect of beam width on the shape of the video signal is shown in Figs. 2.8(a), (b), and (c). Gaussian beam current densities with $\sigma = 1$, 2, 3, and 4 mosaic elements are used. All densities are normalized to constant beam current,

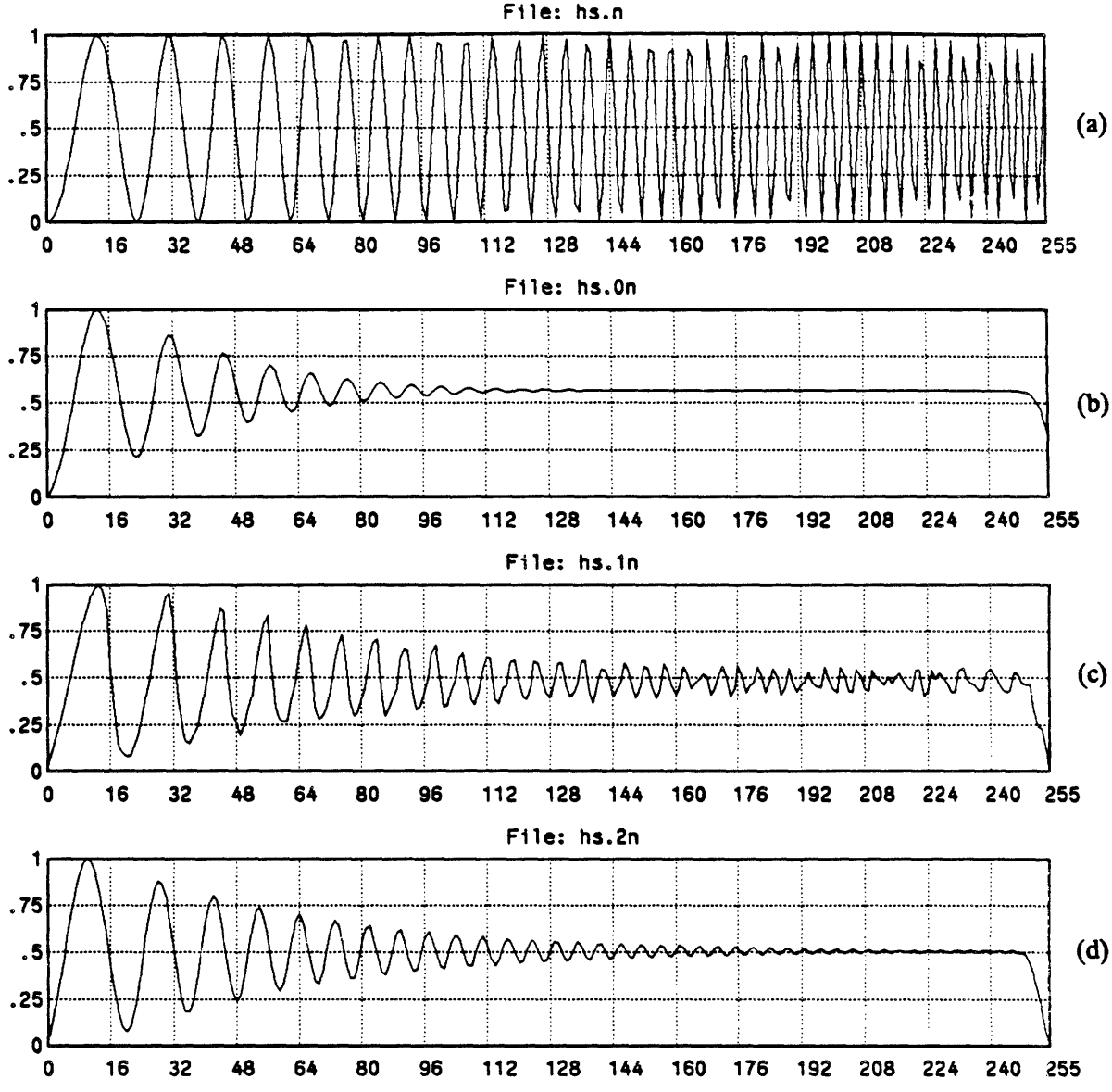


Fig. 2.6. Simulation of the beam sharpening effect. (a) Original charge image. (b) Passive filtering. (c) Linear erasure. (d) Exponential erasure.

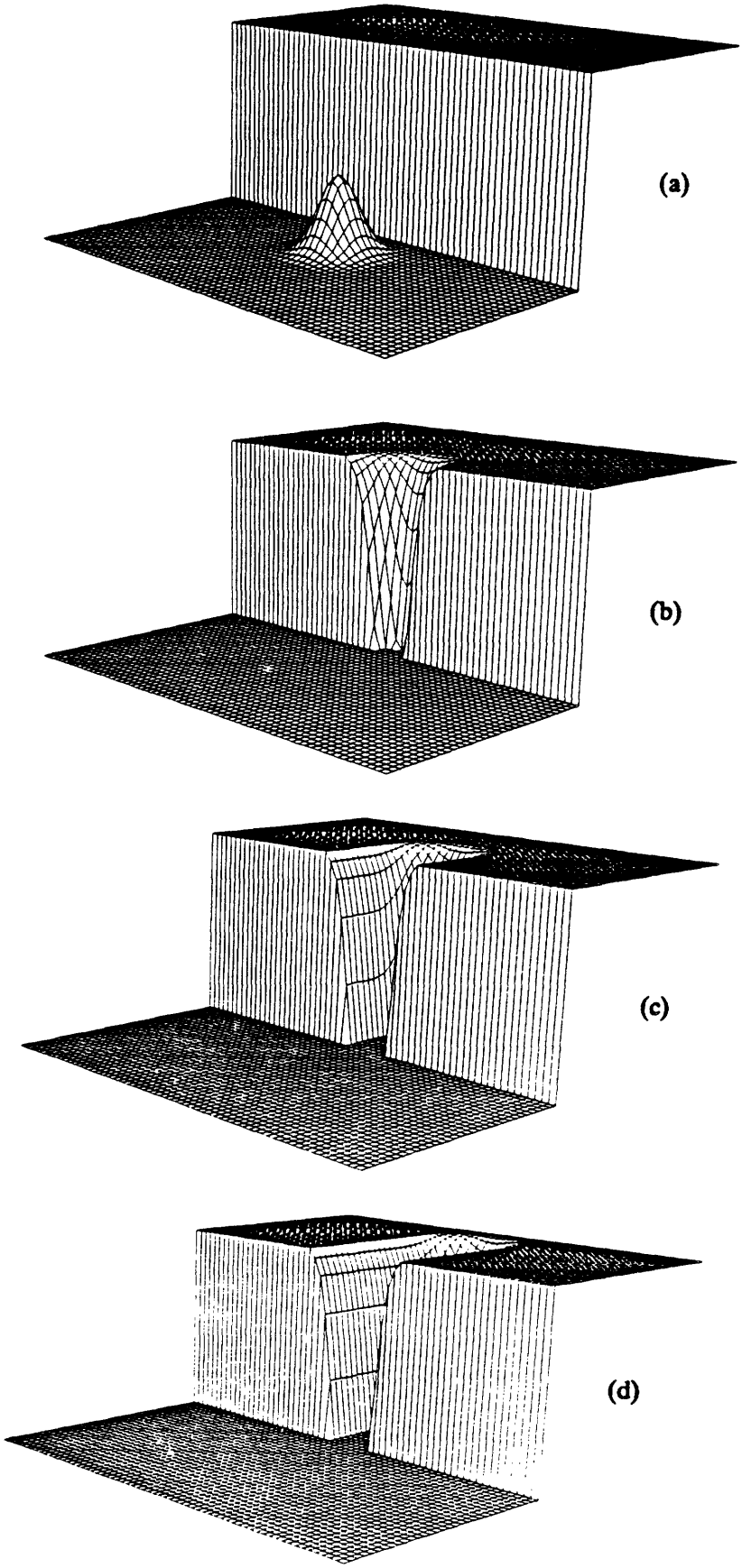


Fig. 2.7. Perspective snapshots of linear charge erasure. (a) Gaussian electron beam about to enter wall of positive charge. (b)-(d) Remaining charge at three different beam positions.

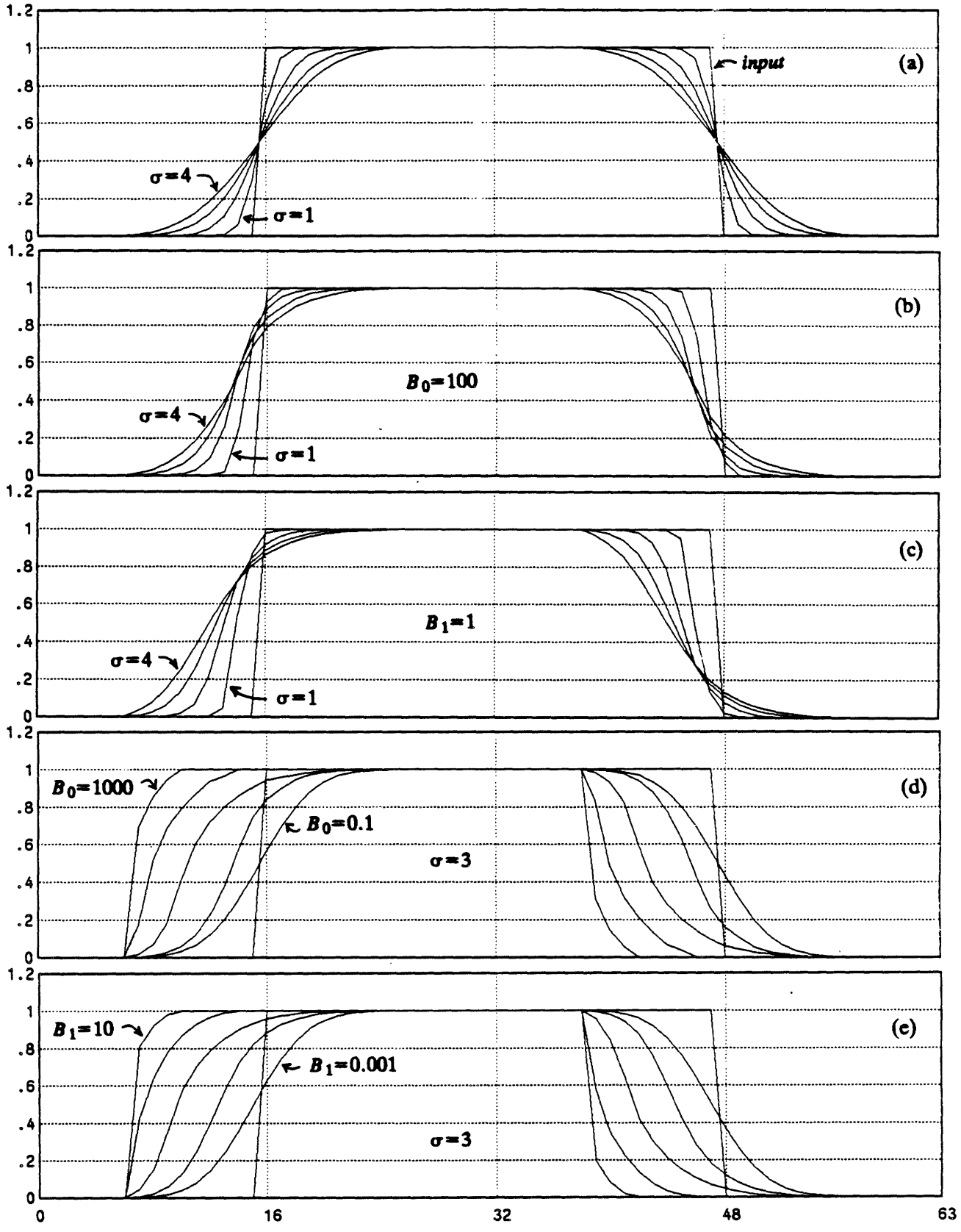


Fig. 2.8. Effect of beam width and amplitude on rise and lead time for constant charge amplitude ($A = 1$). (a) Passive filtering. (b), (d) Linear erasure. (c), (e) Exponential erasure.

and the video signals are normalized to constant peak amplitude. In Fig. 2.8(a), passive filtering is shown. As expected from the zero-phase characteristic of the beam, the rising and falling edges are symmetric with respect to the half-amplitude point of each edge. In Figs. 2.8(b) and (c), linear and exponential erasure are shown. In all cases, the output current *leads* the charge distribution. Furthermore, both the lead time and the 10-90% rise time are proportional to the width of the beam.

Figs. 2.8(d) and (e) show the effect of beam current on the shape of the video signal for linear and exponential erasure. It is assumed that increasing the beam current affects only the intensity of the beam; the width is kept constant at $\sigma = 3$ mosaic elements. The beam amplitude is changed by five orders of magnitude. As the beam current increases, lead time increases and the 10-90% rise and fall times decrease. This happens because a larger beam current allows the *leading edge* of the beam to neutralize more charge.

Fig. 2.9 illustrates the effect of charge amplitude on rise and lead time for constant beam parameters ($B_0 = B_1 = 1$, $\sigma = 3$). Figs. 2.9(a) and (b) show the normalized response of linear and exponential readout when the amplitude of the strip of charge is varied over five orders of magnitude. For linear erasure, rise time decreases and lead time increases as the charge amplitude decreases. However, exponential readout is insensitive to changes in charge amplitude. This is because the beam is always reading out the same *fraction* of charge. After normalization to unit peak amplitude, the absolute signal values are transformed into identical relative values. Figs. 2.9(c) and (d) show a similar effect when the charge image is an impulse centered along a scanning line. Linear erasure, Fig. 2.9(c), shows a strong increase of lead time with decreasing charge amplitude. Exponential erasure, Fig. 2.9(d), shows no dependence of lead time on beam amplitude.

Effect of Beam Amplitude on Image Detail: The linear model can simulate the steady-state effects of insufficient beam current. In real photoconductors, a low beam current leads to target stablization near the saturation limit of the sensor. The result is an image with few highlight details and an overall washed-out appearance. This effect can be modeled by linearly erasing a single frame of charge by a low-

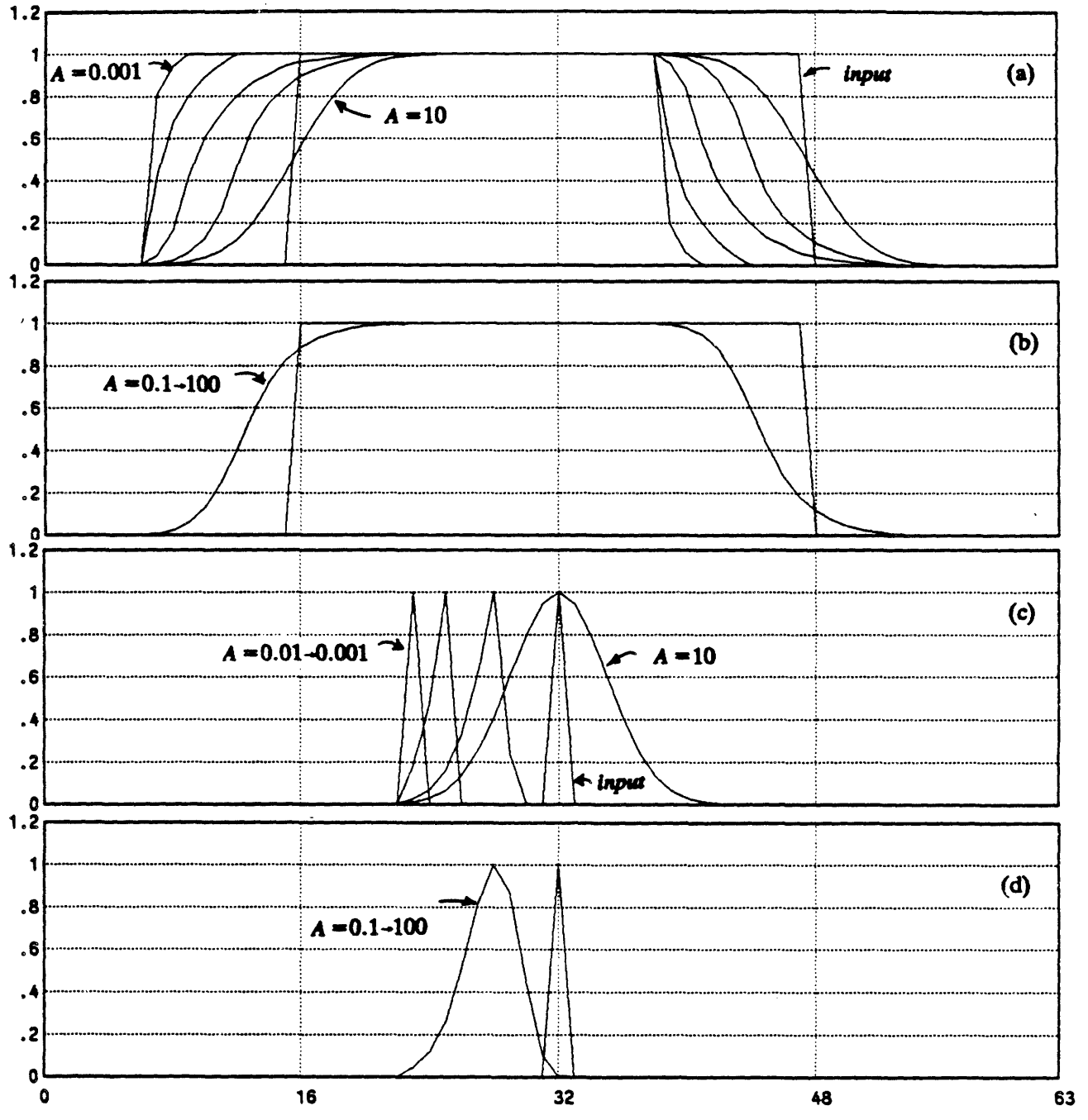


Fig. 2.9. Effect of charge amplitude on rise and lead time for constant beam parameters ($B_0=B_1=1$,$\sigma=3$). (a), (c) Linear erasure. (b), (d) Exponential erasure.

amplitude read beam. Figs. 2.10(a)-(d) show 64-line TV simulations in which Gaussian read beams are used ($\sigma = 0.25$ TV lines) with $B_0 = 25$, 50, 75, and 100 charge units. The peak amplitude of the charge image is 255 units, corresponding to the brightest detail. All displayed images are normalized to constant peak amplitude. Note that the amount of highlight detail increases with beam amplitude. The beam amplitude sets a threshold on the image brightness. Below the threshold, the output signal is proportional to the local charge amplitude; above the threshold, the output signal is constant. For this example, a beam amplitude equal to at least half the peak charge amplitude is necessary to render sufficient highlight detail. Fig. 2.11 shows the effect of varying the beam current in a real vidicon tube. The similarity between Figs. 2.10 and 2.11 indicate the usefulness of the linear erasure model in predicting image appearance at low beam currents[7].

2.3.3.2. Solid-State Cameras

Generally, sensors in solid-state cameras are arranged in a rectangular array containing $N_H \times N_V$ resolution cells. Each cell is of size $H_C \times V_C$ and contains an active light-sensitive area of size $H_A \times V_A$. This structure is shown in Fig. 2.12.

The total charge accumulated in the cell at location (j,k) is given by

$$q_T(j,k) = \int_{A(j,k)} \int q_I(x,y)\,dx\,dy \ , \tag{2.10}$$

where the integration is performed over the active sensing area $x,y \in A(j,k)$ enclosed by the j,k-th cell and q_I is the time-integrated charge density. When the cell is electronically scanned, the output current from the j,k-th cell is

$$i_R(j,k) = \frac{q_T(j,k)}{T_R} \ , \tag{2.11}$$

[7] In his extensive work with vidicons, Prof. W.F. Schreiber observed clipping in the video signal at reduced beam currents. He believes that linear erasure is a good model for a vidicon's tonal response, but that its lag response is better modeled by a sum of decaying exponentials.

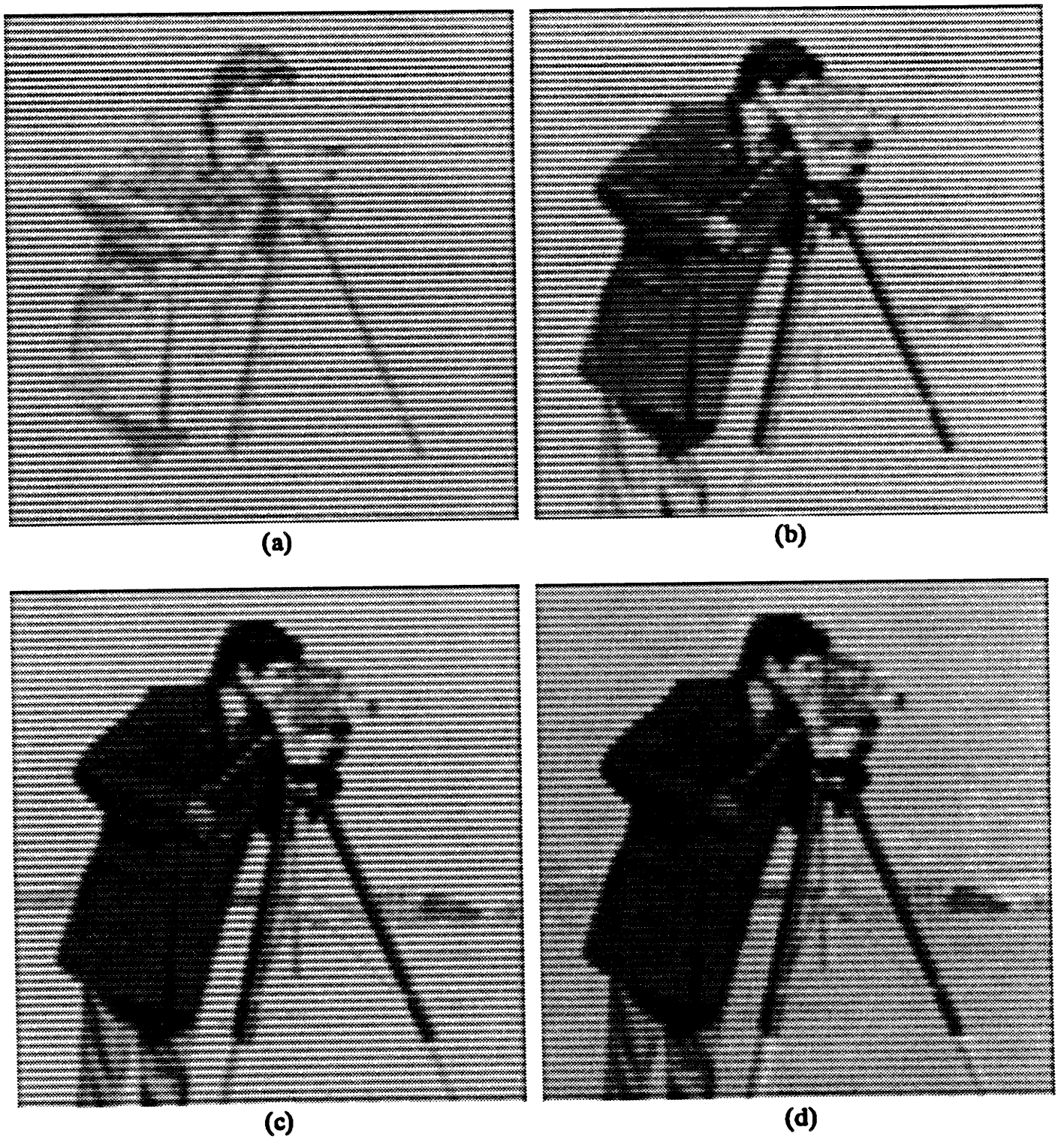


Fig. 2.10. Effect of beam amplitude on image detail using linear erasure. B_0 = (a) 25, (b) 50, (c) 75, and (d) 100 charge units.

(a)

(b)

Fig. 2.11. Effect of beam amplitude on image detail using a vidicon tube. Beam current is low in (a) and is nearly normal in (b).

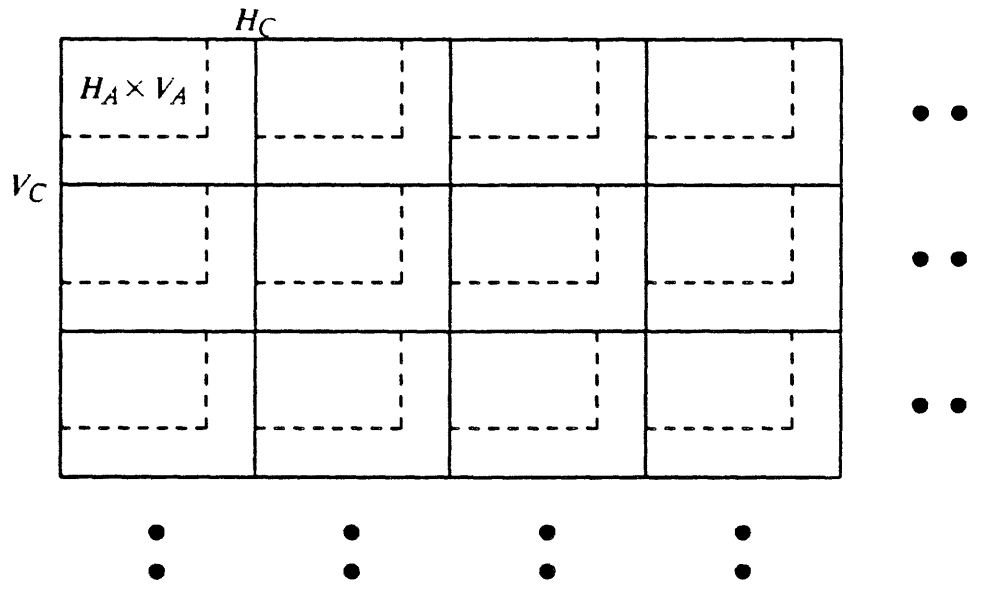


Fig. 2.12. General structure of solid-state array.

where T_R is the readout interval. Conceptually, a 1-D signal is formed by sampling the matrix of currents row-by-row:

$$i_R(t) = i'_R\left(\left[\frac{t}{T_S}\right] \bmod N_H, \left[\frac{t}{N_H T_S}\right] \bmod N_V\right), \tag{2.12}$$

where T_S is the pixel sampling period, $[x]$ is the integer portion of the real number x, and $n \bmod N$ is the integer remainder of n/N.

2.4. Gamma Correction

Picture tubes are high-contrast devices having gamma values in the range of 2-3 [56], with nominal values of 2.2 or 2.8. The overall system gamma is $\gamma_S = \gamma_1 \cdot \gamma_2 \cdot \gamma_D$, where γ_1 is the photoconductor gamma, γ_2 is the external gamma correction at the transmitter, and γ_D is the display gamma. For best results, γ_S

should be in the range 1-1.5, depending on scene content and brightness of surround [56].

Vidicon tubes have $\gamma_1 \approx 0.6$, yielding system gammas near 1.4 with no external gamma correction. However, Plumbicons and CCD's have near-unity gammas, so the video signal must be corrected prior to transmission. The general form is

$$v(t) = r_G \cdot i_R(t)^{\gamma_2} , \tag{2.13}$$

where $v(t)$ is the output video signal in volts, γ_2 is the external gamma correction, and r_G is a proportionality factor in ohms.

2.5. The Channel

The baseband video signal travels through a cable to the display. The channel, as any transmission medium, may degrade the signal. Lowpass filtering, ghosting due to impedance mismatch or multipath reflection, and additive random noise are common channel degradations. A linear time-invariant model that takes into account all three of these degradations is given by

$$v'(t) = v(t) * h_{LPF}(t) * h_{GHOST}(t) + n(t) , \tag{2.14}$$

where $h_{LPF}(t)$ is the lowpass filter impulse response, which may have nonlinear phase characteristics and which may contain a channel attenuation factor, $h_{GHOST}(t)$ is the ghosting impulse response (modeled by an impulse train), and $n(t)$ is additive random noise. By setting $h_{LPF}(t) = h_{GHOST}(t) = \delta(t)$ and $n(t) = 0$, an ideal channel can be modeled.

2.6. The Display

The function of the display is to convert the 1-D video signal into a time-varying 2-D luminance pattern. Interpolation is an inherent property of the display

because instantaneous video signal values must be reproduced as 2-D pixels of finite dimensions. In tube-type displays, vertical interpolation is performed by the finite width of the write beam. In solid-state displays, the video signal is sampled and each sample is reproduced as a small square pixel that either emits or transmits light.

In this section, the display will operate in a hypothetical *write-store* mode, which is described as follows:

(1) The screen is blanked at $t = 0$.

(2) A complete frame of the image is scanned onto the display.

(3) The optical image is stored (like a waveform on an oscilloscope), and the static frame is used for subsequent analysis.

This mode of operation is necessary to produce an intelligible "snapshot" at a certain time instant. The display of a normal CRT, if sampled instantaneously, would consist of a luminous horizontal streak or band, the rest of the image having decayed to zero luminance.

2.6.1. Display Gamma

Tube-type displays have a nearly square-law relationship between cathode current and screen luminance. In general, the relationship is

$$i_G(t) = g_G \cdot v'(t)^{\gamma_D} , \tag{2.15}$$

where γ_D is the display gamma, and g_G is a scale factor.

2.6.2. Charge-to-Light Conversion

The 1-D current waveform must now be converted to luminance information on the display screen. In CRT's, this is performed by an electron beam hitting a phosphor coated screen. In solid-state devices, the current drives a light-emitting

device or regulates a light valve.

The scanning operation maps the 1-D video signal onto a 2-D raster pattern. In CRT's, this mapping is given by

$$i'_G(x,n\Delta y) = i_G\left(\frac{x}{H_D}\cdot T_H + \frac{n\Delta y}{V_D}\cdot T_V\right), \tag{2.16}$$

where $H_D \times V_D$ is the size of the display, T_H and T_V are the horizontal and vertical scanning periods, and $0 \le x \le H_D$, $0 \le n\Delta y \le V_D$ are the xy-coordinates on the face of the display, with the origin in the upper left corner. Note that the mapping from 1-D to 2-D is only defined for discrete values of y, corresponding to horizontal scan line centers. A similar discretized version describes the mapping onto the output grid of a solid-state display. The general structure of a solid-state display is the same as that of a solid-state camera (Fig. 2.12).

The charge-to-light conversion process can be modeled quite well by the linear equation

$$l_D(x,y) = C_D \cdot i'_G(x,y) * h_D(x,y) + L_D , \tag{2.17}$$

where $l_D(x,y)$ is the displayed luminance pattern, $i'_G(x,y)$ is the current at location (x,y), $h_D(x,y)$ is the impulse response of the display, C_D is a contrast scaling factor, and L_D is a luminance offset. For CRT's, $h_D(x,y)$ is usually approximated by a circular Gaussian spot. Since $i'_G(x,y)$ is defined only at discrete values of y, the convolution simulates the line structure seen on monochrome CRT's. For solid-state displays, $h_D(x,y)$ is usually a uniform rectangular function, and $i'_G(x,y)$ is discrete in both spatial dimensions.

2.7. The Human Visual System

The human visual system is the ultimate processor of displayed information. Many models exist for the spatial processing of the HVS. Several mathematical

models used to describe the nonlinear characteristics of the eye are [56]

$$b_1(x,y) = a_1 \cdot \log(l_D(x,y)) + k_1 \,, \tag{2.18a}$$

$$b_2(x,y) = \frac{a_2(l_D(x,y) - l_T)^{\gamma_{HVS}}}{f_2(l_D(x,y))} \,, \tag{2.18b}$$

$$b_3(x,y) = \frac{a_3 \cdot l_D(x,y) \cdot (L_O + k_2)}{l_D(x,y) + L_O} \,, \tag{2.18c}$$

where a_1, a_2, a_3, k_1, k_2 are appropriate constants, l_T is the threshold luminance, L_O is the surround luminance, and γ_{HVS} is the gamma of the human visual system. $\gamma_{HVS} \approx 1/3$ for dark surround, 1/2 for bright surround.

In general, the spatiotemporal response of the HVS is bandpass in all three dimensions, and the spatial and temporal responses are not completely separable at all frequencies [64]. However, for any given temporal frequency, the spatial frequency response is bandpass in nature. Hall and Hall [65] proposed a lowpass-log-highpass model of the spatial portion of the HVS. A typical spatial frequency response is shown along with its impulse response in Fig. 2.13 [56]. An approximation to the perceived brightness pattern is given by

$$b_D(x,y) = b_n(x,y) * h_{HVS}(x,y) \,, \tag{2.19}$$

where $b_n(\cdot,\cdot)$ is a suitable nonlinear transformation and $h_{HVS}(x,y)$ is the spatial impulse response of the HVS.

2.8. Experiments

To assess the effect of aperture size and shape on image quality in both tube-type and solid-state systems, some experiments were run using TVSIM, a TV simulation program discussed in detail in Chapter 5.

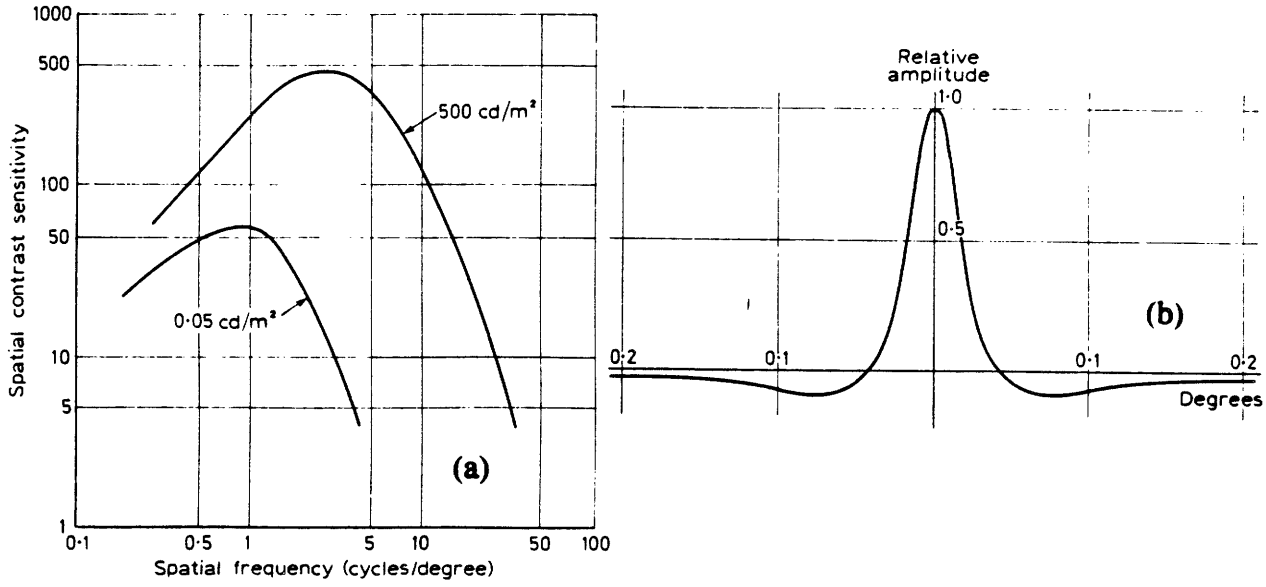


Fig. 2.13. (a) HVS spatial frequency response. (b) HVS spatial impulse response.

2.8.1. Simulation of Tube-Type TV Systems

The following camera and display parameters remained constant throughout the experiments:

Camera Parameters:

Total number of lines per frame: 64

Camera mosaic aspect ratio: 1:1

Charge readout mode: Passive linear filtering.

Photoconductive gamma: 1.0

External gamma: 1.0

Scan line pitch: 4 mosaic elements

Interlace ratio: 1:1 (progressive scan)

Fraction of line time for horizontal blanking: 0.0

Fraction of frame time for vertical blanking: 0.0

Display Parameters:

Total number of lines per frame: 64

Display mosaic aspect ratio: 1:1

Display gamma: 1.0

Scan line pitch: 4 mosaic elements

Interlace ratio: 1:1 (progressive scan)

Fraction of line time for horizontal blanking: 0.0

Fraction of frame time for vertical blanking: 0.0

Contrast adjustment: None

Brightness adjustment: None

2.8.1.1. Effect of Read and Write Beam Width on Image Quality

In this experiment, circular Gaussian spots characterize the camera and display tubes. Both are assumed linear, so the entire system is governed by the two parameters σ_r and σ_w, the read and write beam sigmas, which are varied in increments of 0.1 TV lines (TVL). The width of a TV line is equal to the scan line pitch, or vertical separation between scan line centers. σ_r ranges from 0.2 to 0.4 TVL, and σ_w ranges from 0.3 to 0.5 TVL. Smaller values of σ_r are chosen because moderate aliasing is permitted in most commercial TV systems. The resulting set of nine pictures is shown in Fig. 2.14. As σ_r increases, so does blur, but aliasing is reduced; as σ_w increases, so does blur, but the scanning lines become less visible. For this particular image, the most pleasing result occurs at $\sigma_r \approx 0.2$ TVL and $\sigma_w \approx 0.4$ TVL. This same experiment is repeated for a computer-generated test pattern, and the results are shown in Fig. 2.15.

At these values of beam width, the read beam causes some aliasing, which for natural imagery is not objectionable; the write beam produces visible scan lines, which is less objectionable than too much blur. In TV studios, as well as in most homes, the display beam is often adjusted for best detail at the expense of increased line visibility [66].

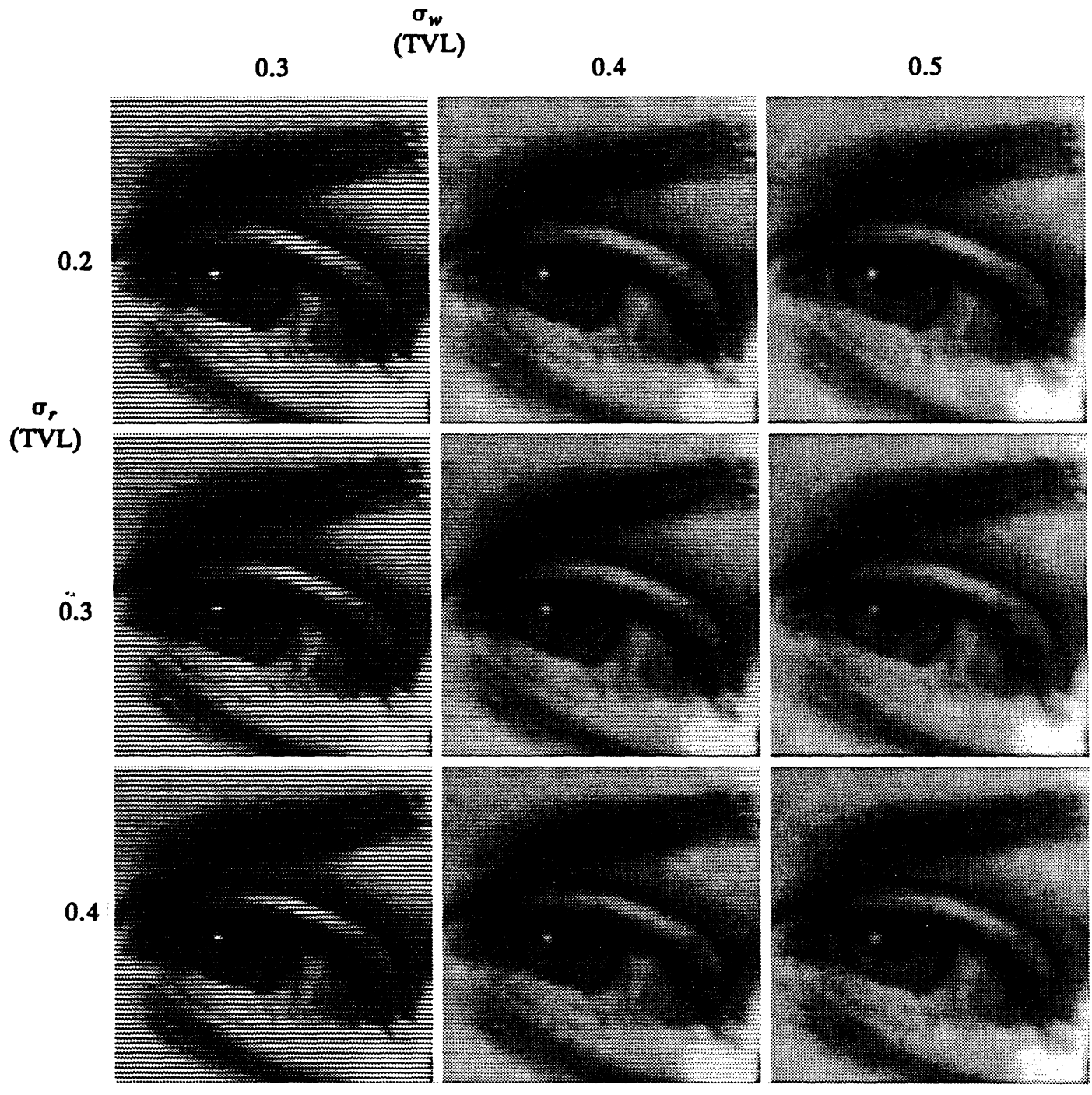


Fig. 2.14. Effect of Gaussian beam widths on image quality. Natural image.

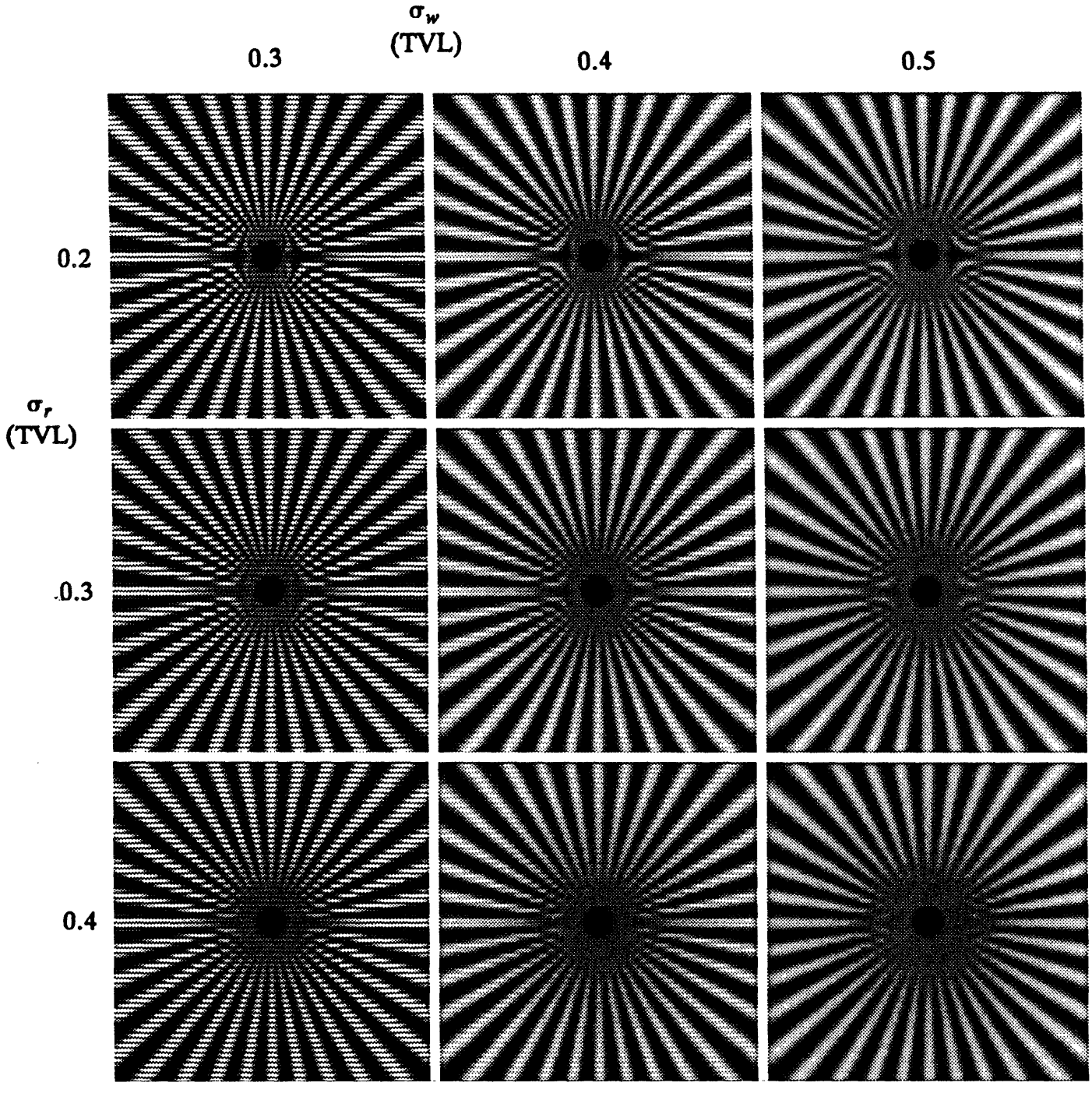


Fig. 2.15. Effect of Gaussian beam widths on image quality. Test pattern.

This experiment is similar to that performed Ratzel [47], and the results are also similar. Ratzel isotropically filtered, subsampled, and interpolated in *both* spatial directions and found optimal prefilter and postfilter widths to be 0.3 and 0.375 subsample spacings, respectively.

2.8.1.2. 2-D Frequency-Domain Analysis of Video Systems

In this section, the effect of passive tube-type scanning is examined in the frequency domain. The experiments simulate a 64-line TV system in which the shape and size of the read beam and write beam filters are the only parameters. The input picture is a standard cameraman test picture, (size 256H×256V).

The series of images in Fig. 2.16 illustrate the final output from each of the experiments. Beneath each spatial-domain image is shown the log magnitude of its 2-D Fourier transform (DC is at the center of the spectrum). Observing the effects of various filtering operations in the 2-D spectrum is useful in analyzing the results. A visual correspondence between blurring and aliasing can be obtained in both domains. In the Fourier transform pictures, 0 dB is shown as white, -48 dB is shown as black, and the difference between grey levels is 3/16 dB. Components below -48 dB are clipped to the minimum black level.

Each paragraph below refers to the corresponding picture/transform pair in the accompanying set of figures:

- **Fig. 2.16(a). Original (256H×256V)**

  The original photograph is sampled at 722/16 ≈ 45 pixels/inch on an ECRM 8400 Autokon. It is cropped to a size of 256×256, implying a maximum vertical spatial frequency of 256/2 = 128 cycles/picture height (cph). This sampling rate is not high enough to avoid aliasing, and jagged edges are visible along the tripod legs. This artifact corresponds to the two lines in the NW and SE quadrants of the spectrum. The other bright, long spectral lines correspond to high-contrast, thin, elongated spatial objects whose major axis is perpendicular to the direction of the spectral line. Except for the bright spectral lines, most of the energy in the

spectrum is concentrated near DC, the bright point in the center of the spectrum. This is typical of most natural imagery.

- **Fig. 2.16(b). Read Beam: impulse; Write Beam: impulse**
  Here a beam spot whose dimensions are one mosaic element (effectively an impulse) is used to record and display the image. This is equivalent to subsampling the image vertically by a factor of 4 (the line pitch), and then zero-padding by the same factor. In the frequency domain, this results in a vertical replication of the baseband spectrum at the frequencies $2\pi k/4$ ($64k$ cph), where k is an integer. The energy in each replicated spectrum is equal to that in the baseband; this is *aliasing without blurring*. The line structure in the reproduced image is very visible and is not suitable for prolonged viewing.

- **Fig. 2.16(c). Read Beam: impulse; Write Beam: 4H ⁂ 4V Prism**
  Here most of the energy of the alias spectra is set to zero by the transform of the write beam filter. Line structure is not visible in uniform areas of the picture, but jagged edges are seen along sloping high-contrast objects. In the spectral domain, this artifact appears as bright sloping lines in the vertical high frequency areas. This experiment simulates the general class of raster-scan output devices with square, uniform apertures.

- **Fig. 2.16(d). Read Beam: impulse; Write Beam: 1H ⁂ 7V Triangle**
  In this experiment, linear interpolation is performed vertically at the receiver. This reproduces uniform areas of the picture perfectly, but performs poorly at sloping high-contrast edges. In the spectrum, the strongest alias components are eliminated.

- **Fig. 2.16(e). Read Beam: impulse; Write Beam: 7H ⁂ 7V Pyramid**
  Here the write beam has a pyramid shape; it filters horizontally and linearly interpolates vertically. It looks more blurred but more natural than Fig. 2.16(d). Since the image has already been aliased vertically by the read beam operation, the only way to diminish the visibility of the aliased components at the receiver is

to blur the received image. Notice, however, that there are still remnants of alias components in the spectrum. This appears as blurred staircasing at high-contrast edges in the image.

- **Fig. 2.16(f). Read Beam: 1H × 23V $\frac{\sin x}{x}$; Write Beam: 1H × 23V $\frac{\sin x}{x}$**

  This experiment is non-physical and is interesting only in theory. It illustrates the effect of "ideal" sampling and reconstruction of raster scanned images. To avoid aliasing, with a line pitch of 4, the image must be limited to $|\omega_y| \leq \pi/4$ (32 cph) in the vertical dimension. This implies an ideal low-pass presampling filter with a $\pi/4$ (32 cph) cutoff. The interpolation filter has the same characteristics. A truncated $\sin x/x$ for both read and write beams preserves all horizontal detail at the expense of diminished vertical detail and ringing in the vertical direction. It is well known that filtering images with very sharp cutoff frequencies produces undesirable artifacts. This experiment illustrates this phenomenon for the special case of raster scanning.

- **Fig. 2.16(g). Read Beam: impulse; Write Beam: 9H ⁎ 9V Gaussian**

  Here the write beam spot profile is a 2-D Gaussian, which approximates that of a physical display [67, 68]. When Gaussian beams are used as FIR linear filters, their region of support must be large enough to insure negligible truncation error. In practice, it is adequate to limit the filter size in each direction to six times the corresponding sigma value. The sigma values may be expressed in mosaic elements or in TV lines (TVL). When expresses in mosaic elements, σ_H and σ_V may be converted to TVL by dividing by the scan line pitch. In this example, the line pitch is 4 mosaic elements, so $\sigma_H = \sigma_V = 1.5$ display mosaic elements = 0.375 TVL. The line structure is visible in the image. Alias spectra are attenuated, but not completely eliminated; jagged edges are still visible.

- **Fig. 2.16(h). Read Beam: impulse; Write Beam: 11H ⁎ 11V Gaussian**

  The write beam spot size is enlarged, corresponding to a defocused electron beam in the display. In this experiment, $\sigma_H = \sigma_V = 1.83$ display mosaic elements =

0.458 TVL. Line structure is still visible, and jagged edges have not been completely eliminated.

- **Fig. 2.16(i). Read Beam: impulse; Write Beam: 13H ⁎ 13V Gaussian**
  The write beam spot size is enlarged further. Here $\sigma_H = \sigma_V = 2.17$ display mosaic elements = 0.542 TVL. Line structure is much less visible, and jagged edges have been nearly eliminated. This choice of display sigma is slightly less than 0.57 TVL, which is the minimum theoretical width required to render a flat field with no luminance modulation [68].

- **Fig. 2.16(j). Read Beam: impulse; Write Beam: 15H ⁎ 15V Gaussian**
  The write beam spot size is enlarged further. Here $\sigma_H = \sigma_V = 2.5$ display mosaic elements = 0.625 TVL. Line structure is almost completely eliminated, at the expense of a very blurry picture. The alias components are so severely attenuated at this point that the spectrum is almost equivalent to that of the original picture filtered to $\pi/4$ (32 cph) with a circularly-symmetric Gaussian.

- **Fig. 2.16(k). Read Beam: impulse; Write Beam: 7H ⁎ 15V Gaussian**
  The previous experiment showed that when $\sigma_V = 0.625$ TVL, line structure is nearly invisible even at close inspection. Keeping σ_V constant, σ_H is decreased to 0.292 TVL, producing a Gaussian write beam with an elliptical cross section. This enhances horizontal detail, but aliasing becomes more apparent.

- **Fig. 2.16(l). Read Beam: impulse; Write Beam: 3H ⁎ 15V Gaussian**
  If vertical aliasing has already been introduced into the sampled image, an elliptical display spot with $\sigma_H << \sigma_V$ will enhance some of the horizontal detail at the expense of severe staircasing artifacts at high-contrast, sloping edges. Here $\sigma_H = 0.125$ TVL and $\sigma_V = 0.625$ TVL.

- **Fig. 2.16(m). Read Beam: 3H × 3V Gaussian; Write Beam: 3H ⁎ 15V Gaussian**
  To diminish aliasing effects, the input should be filtered before (or while) it is

being sampled. This experiment models the read beam as a circularly-symmetric Gaussian with $\sigma = 0.125$ TVL. The image is more blurred but less aliased than Fig. 2.16(l).

- **Fig. 2.16(n). Read Beam: 3H × 9V Gaussian; Write Beam: 3H ※ 15V Gaussian**

  The read beam is made elliptical, filtering more in the vertical direction in an effort to reduce vertical alias components. Alias artifacts, while visible in the spectrum, are hardly noticeable in the image. However, the horizontal resolution is much higher than the vertical resolution; this is undesirable [69].

- **Fig. 2.16(o). Read Beam: 3H × 15V Gaussian; Write Beam: 3H ※ 15V Gaussian**

  When the read and write beams are elliptical and have the same shape, the image appears to have no aliasing artifacts, but is unnatural in that it has nearly four times higher resolution in the horizontal direction.

- **Fig. 2.16(p). Read Beam: 9H × 9V Gaussian; Write Beam: 3H ※ 15V Gaussian**

  The final experiment simulates what is often done in practice. Here a circularly symmetric read beam filter ($\sigma = 0.375$ TVL) is used. This filter has 95% of its energy within 0.37π (47 cph), so it attenuates alias components very well. The write beam filter is elliptical to remove line structure without unnecessary filtering in the horizontal direction. The slight anisotropy in resolution is not overly objectionable and is certainly more visually pleasing than many of the read beam/write beam combinations discussed above.

2.8.2. Effect of Apertures in Solid-State TV Systems

In this section, a solid-state camera and a solid-state display consisting of 64×64 square resolution cells are used to study the effects of aperture size on image quality. Each resolution cell is spatially discretized to include 4×4 mosaic

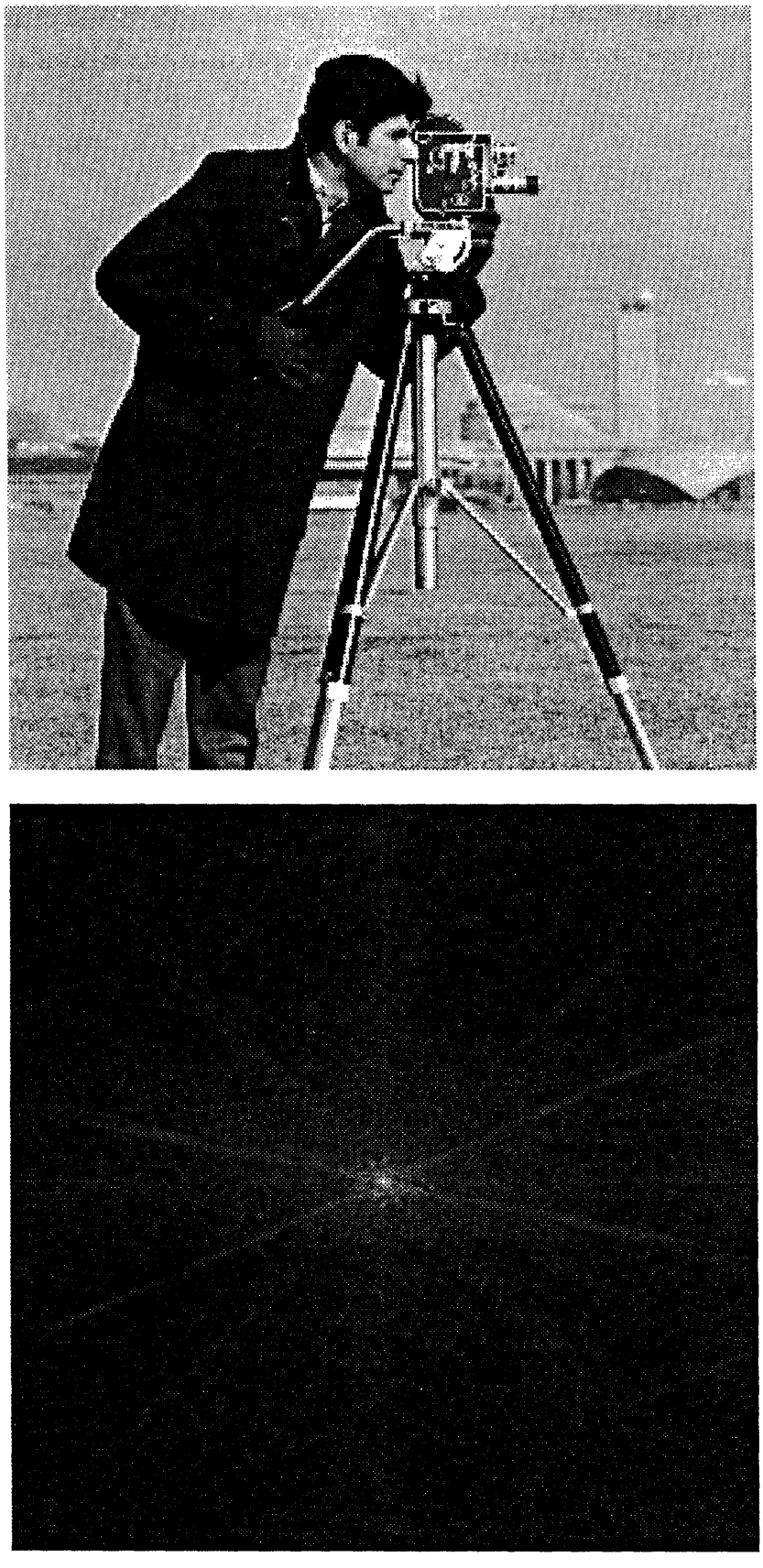


Fig. 2.16(a). CMAN original (256H×256V).

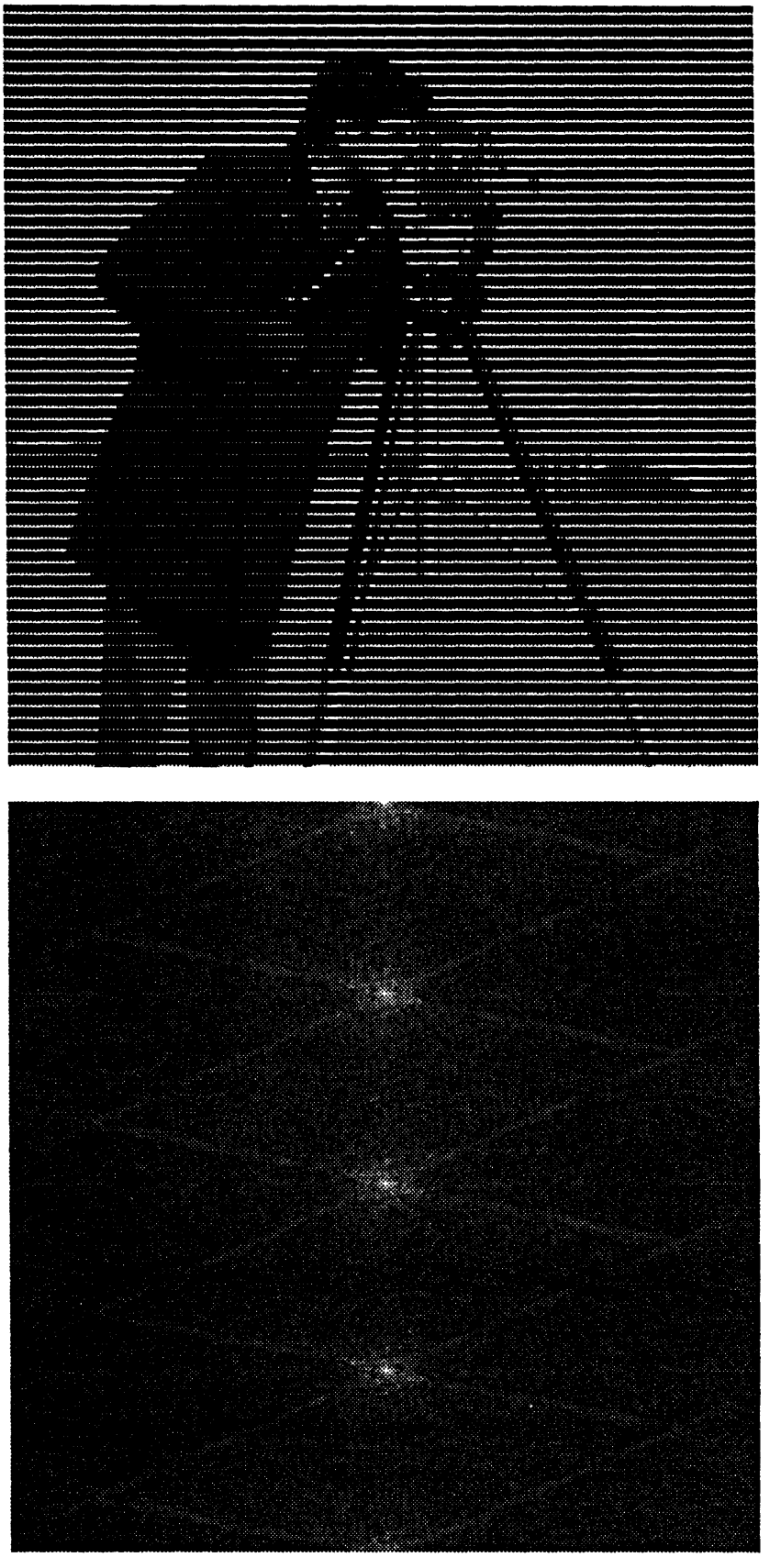


Fig. 2.16(b). Read beam: impulse; write beam: impulse

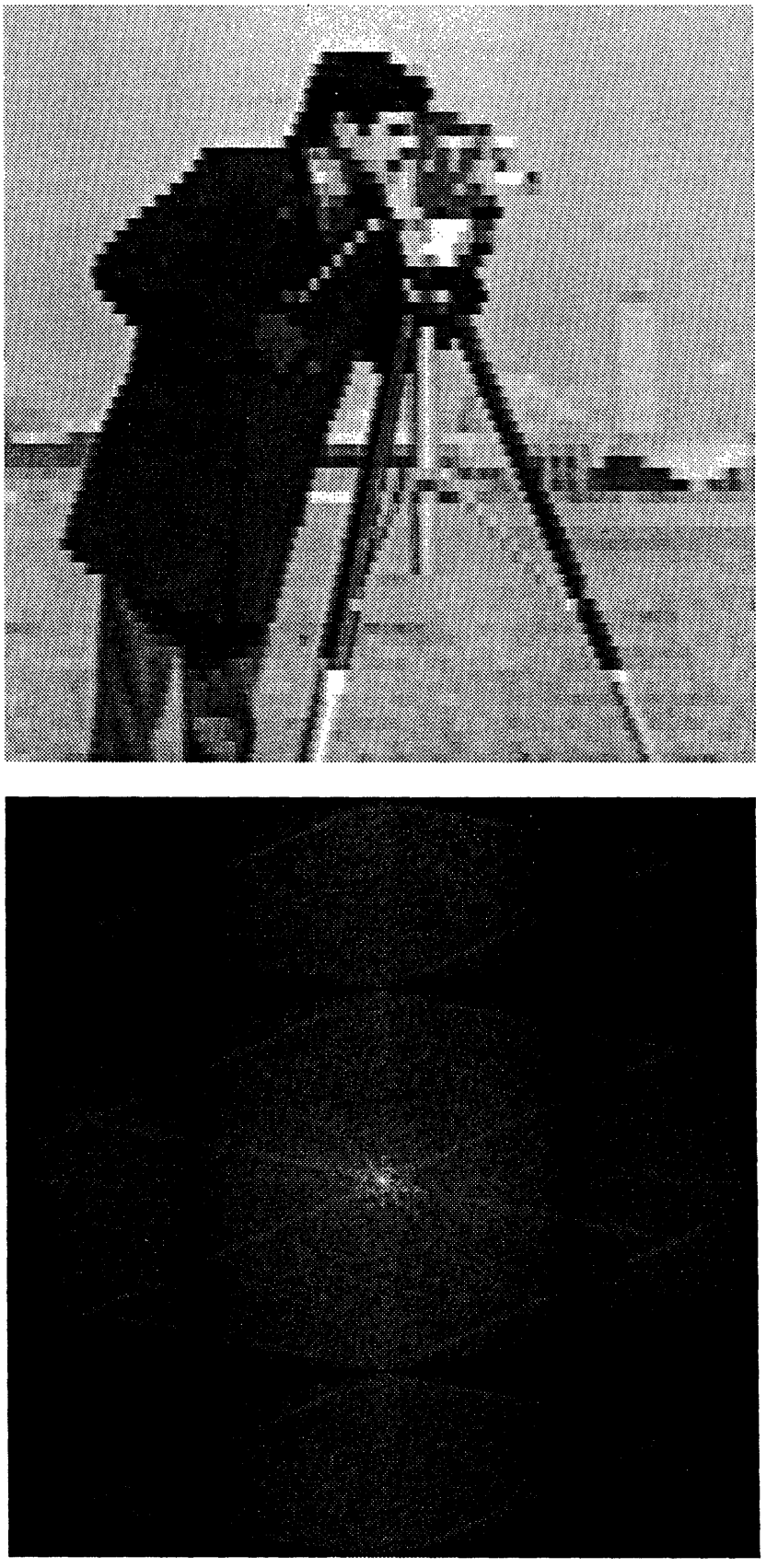


Fig. 2.16(c). Read beam: impulse; write beam: 4H×4V prism

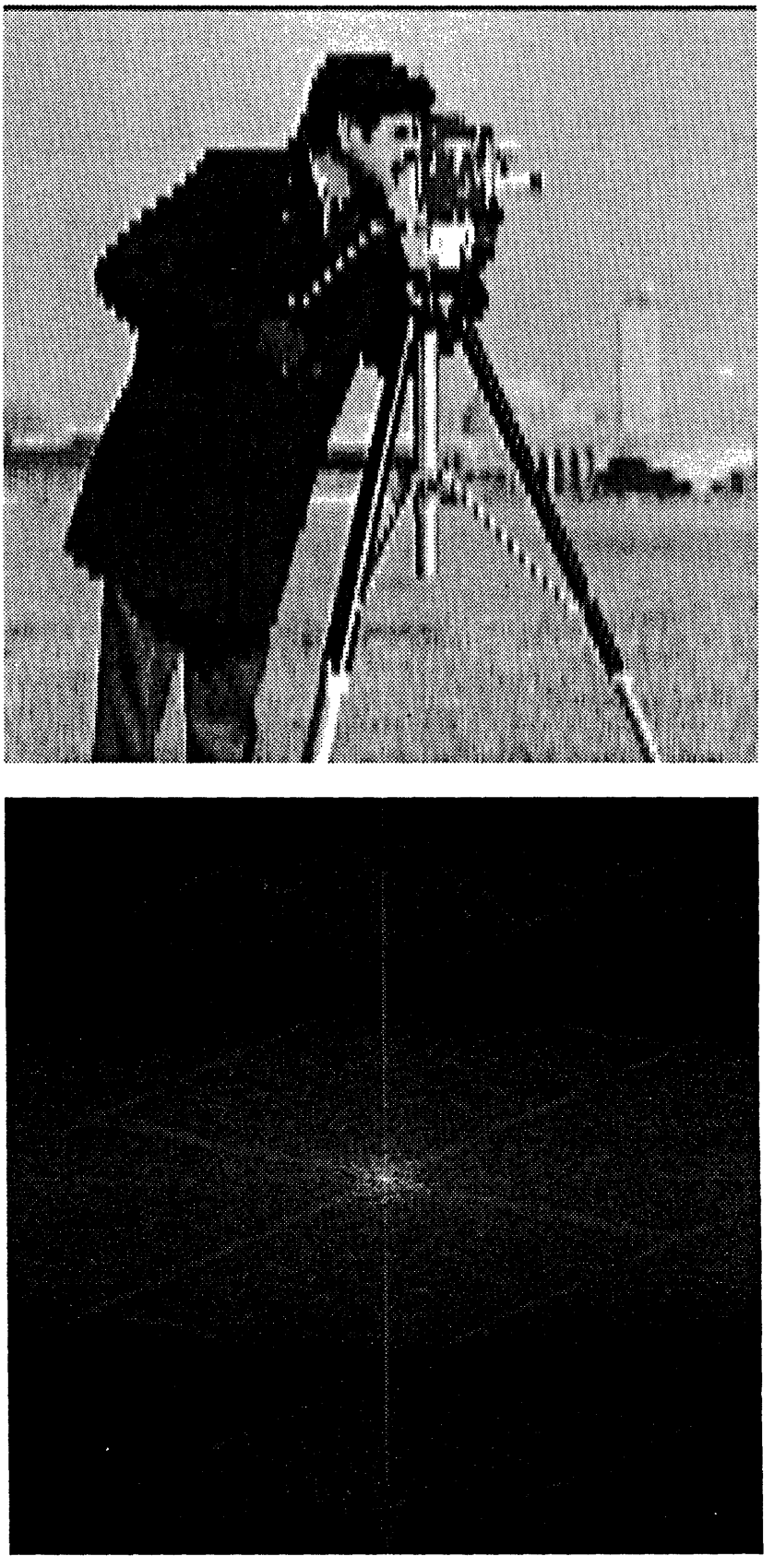


Fig. 2.16(d). Read beam: impulse; write beam: 1H×7V triangle

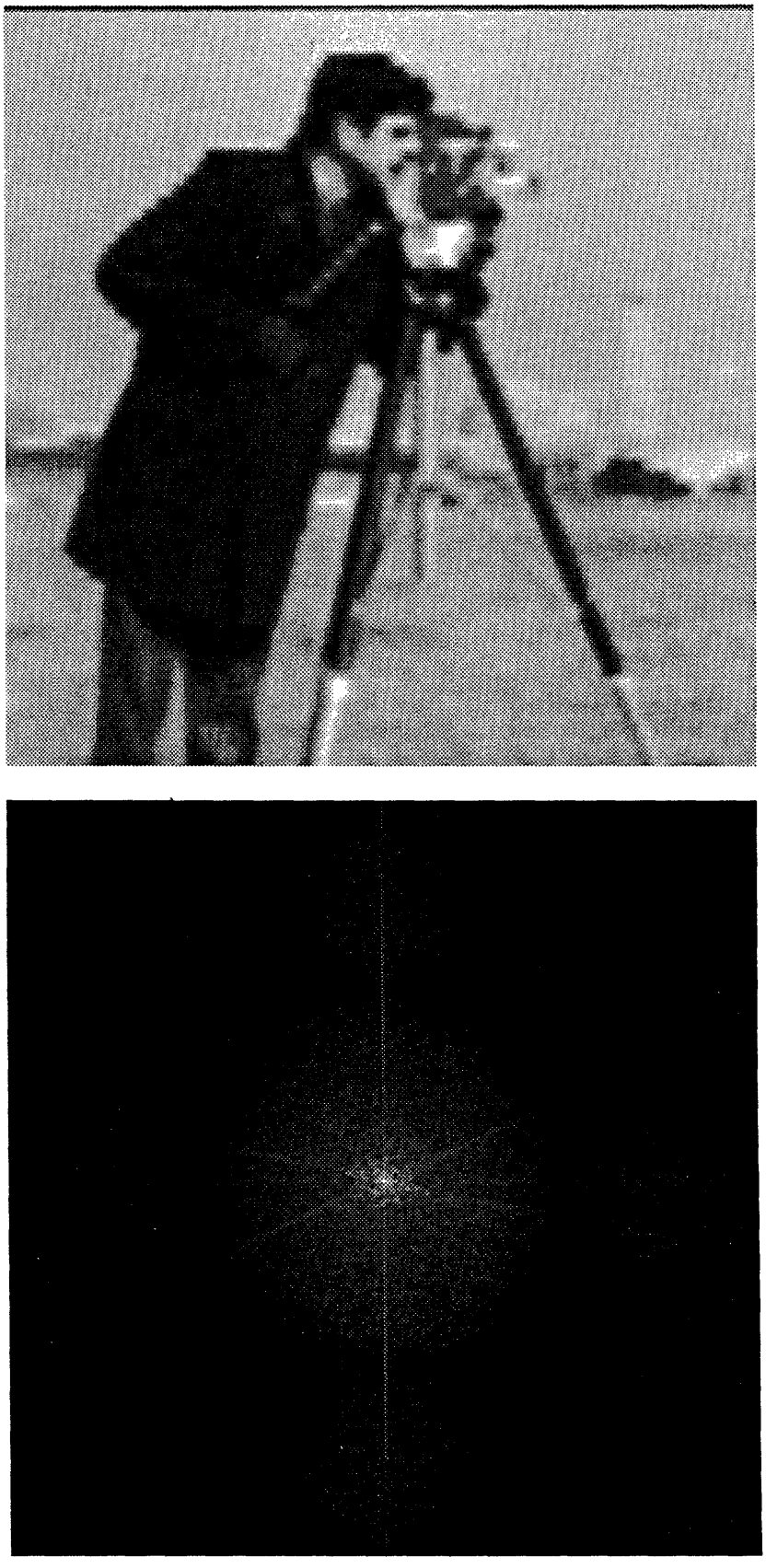


Fig. 2.16(e). Read beam: impulse; write beam: 7H×7H pyramid

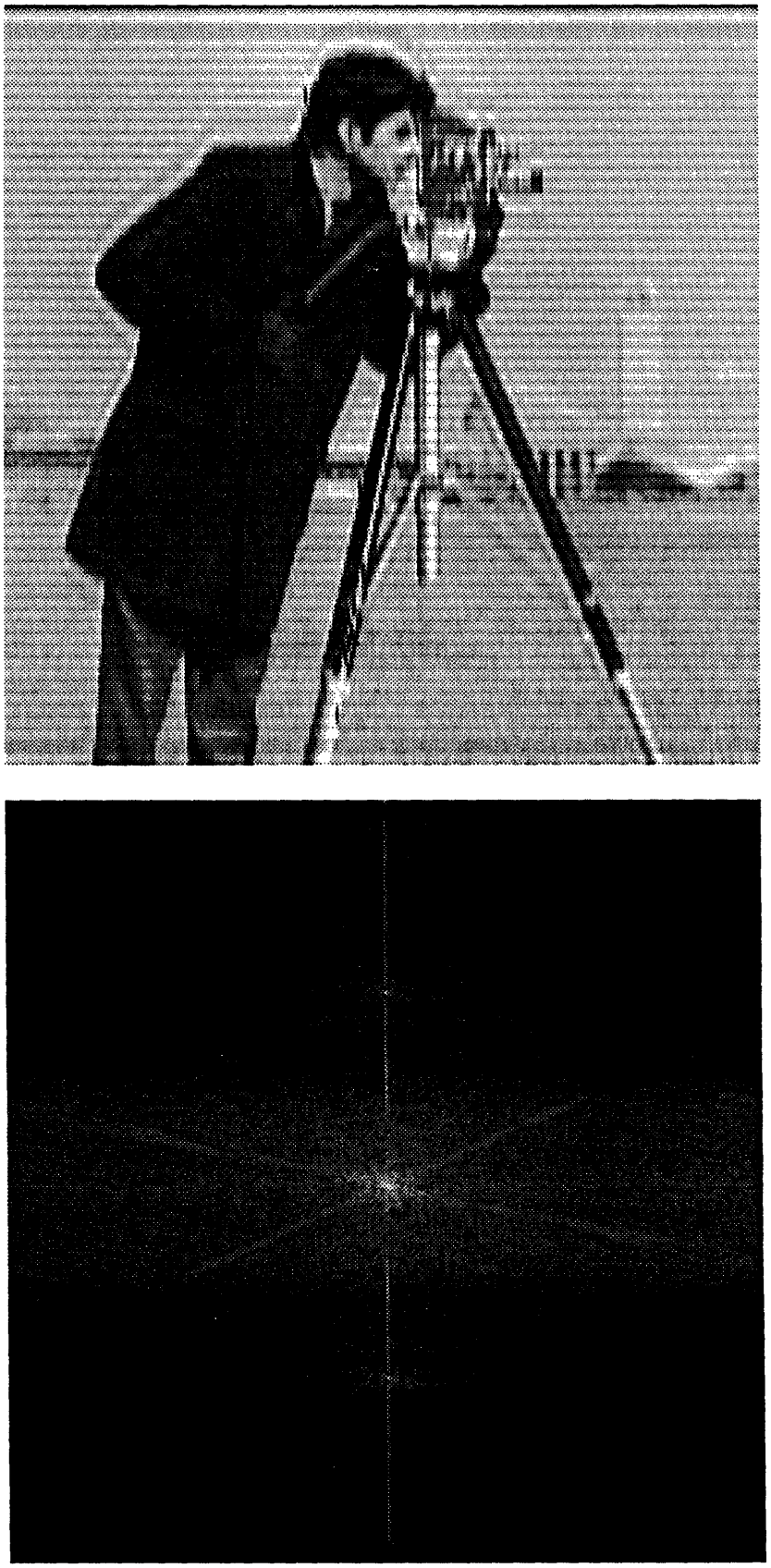


Fig. 2.16(f). Read beam: 1H×23V $\frac{\sin x}{x}$; write beam: 1H×23V $\frac{\sin x}{x}$

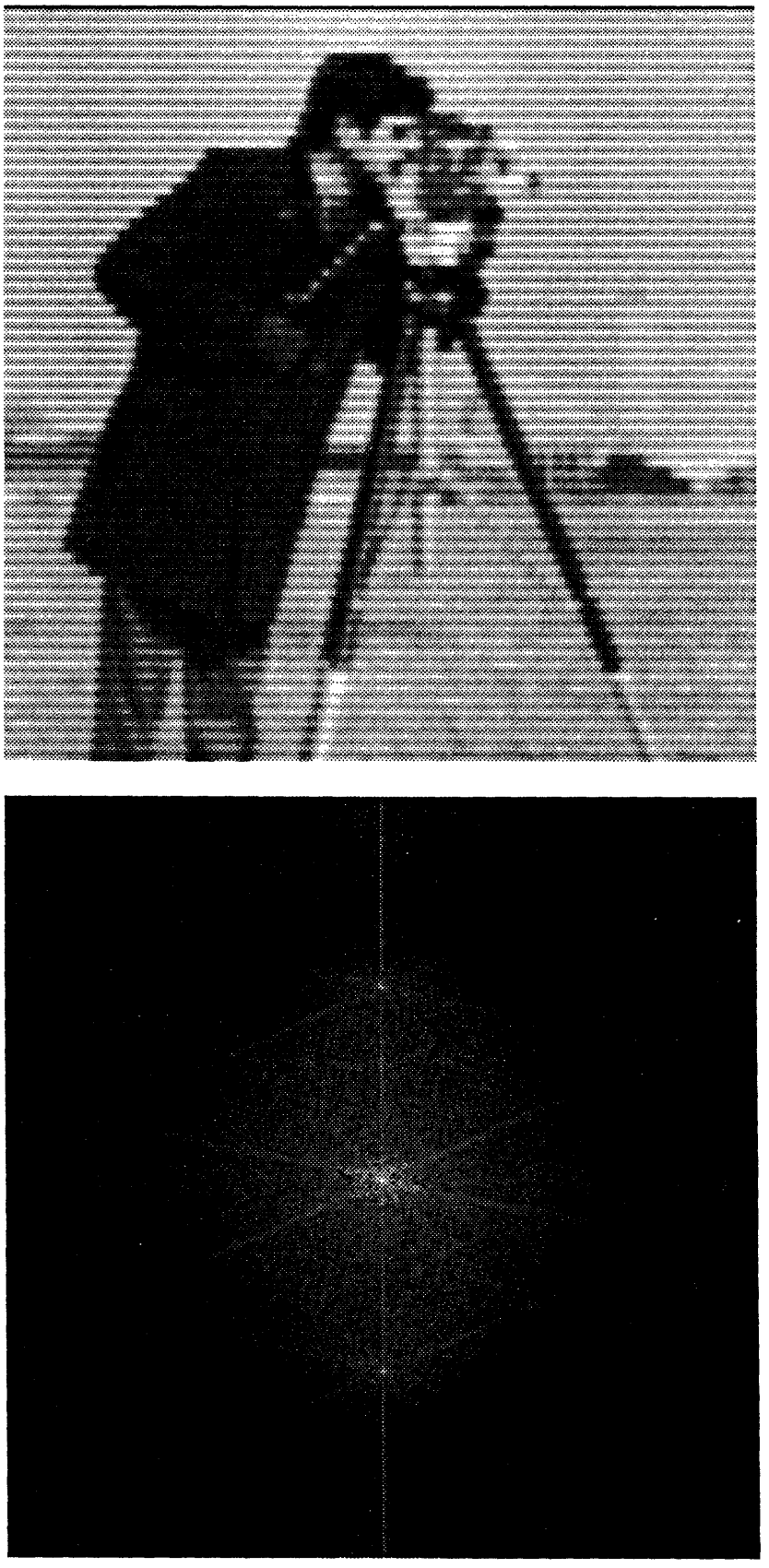


Fig. 2.16(g). Read beam: impulse; write beam: 9H×9V Gaussian

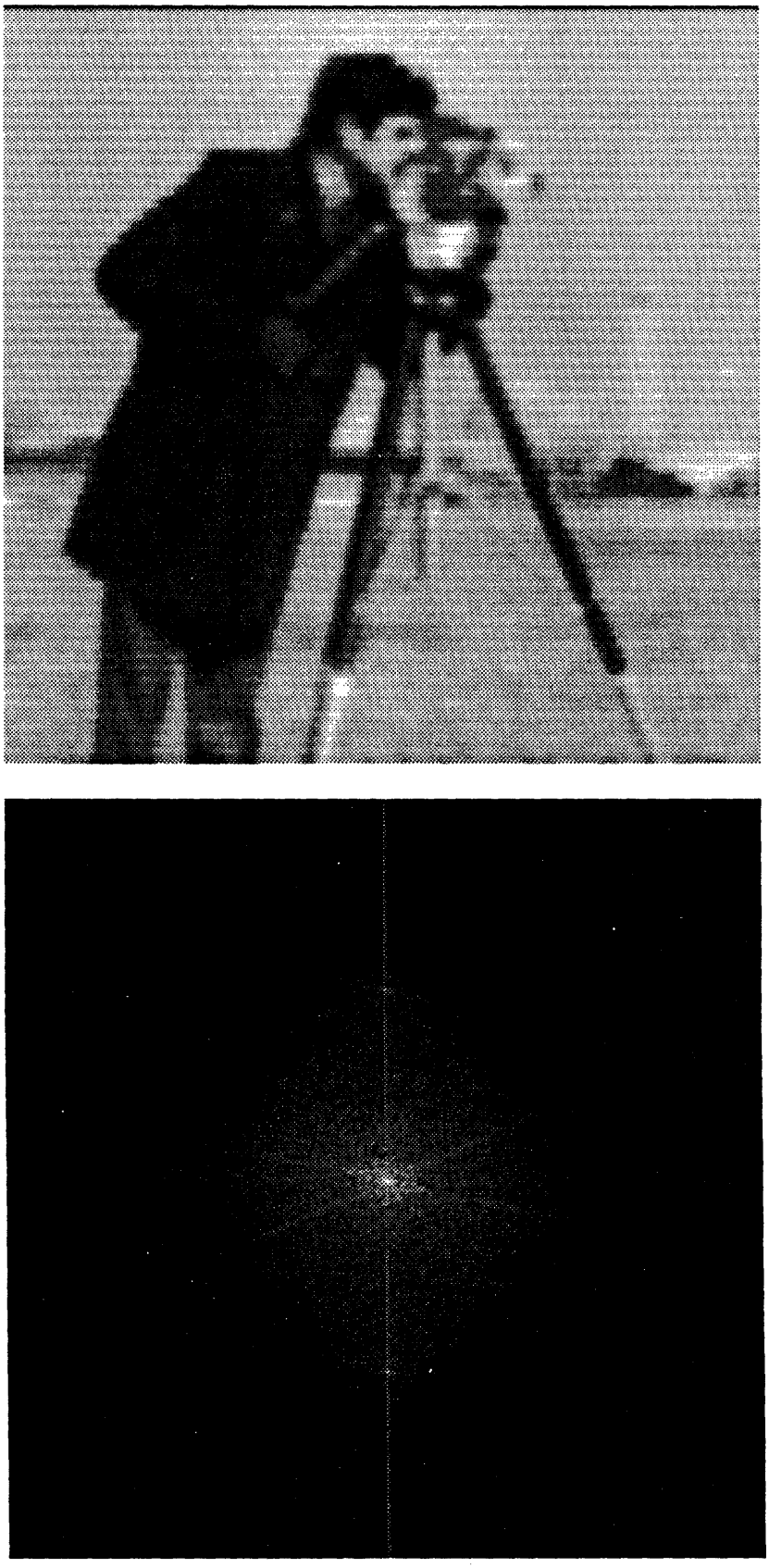


Fig. 2.16(h). Read beam: impulse; write beam: 11H×11V Gaussian

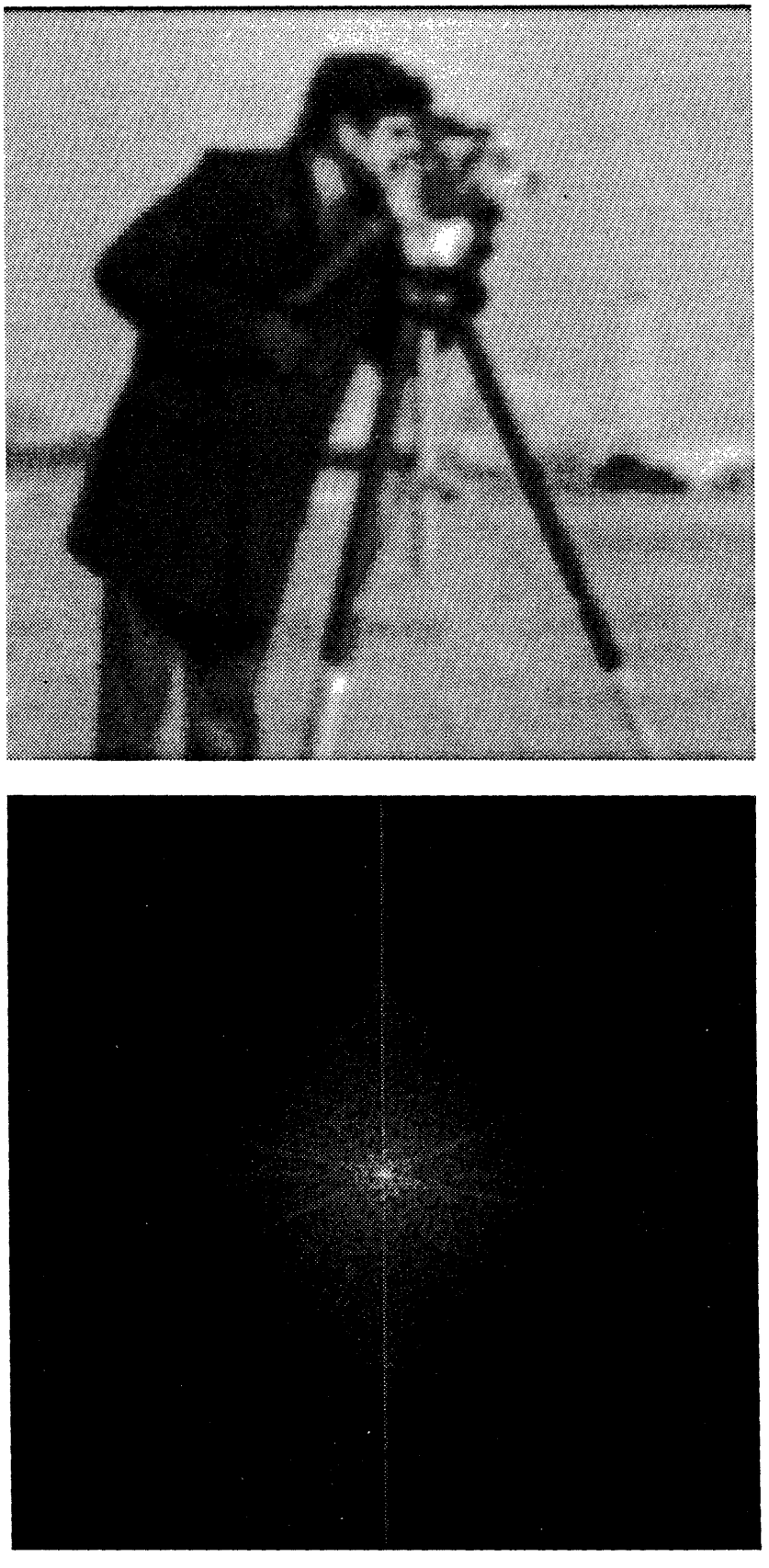


Fig. 2.16(i). Read beam: impulse; write beam: 13H×13V Gaussian

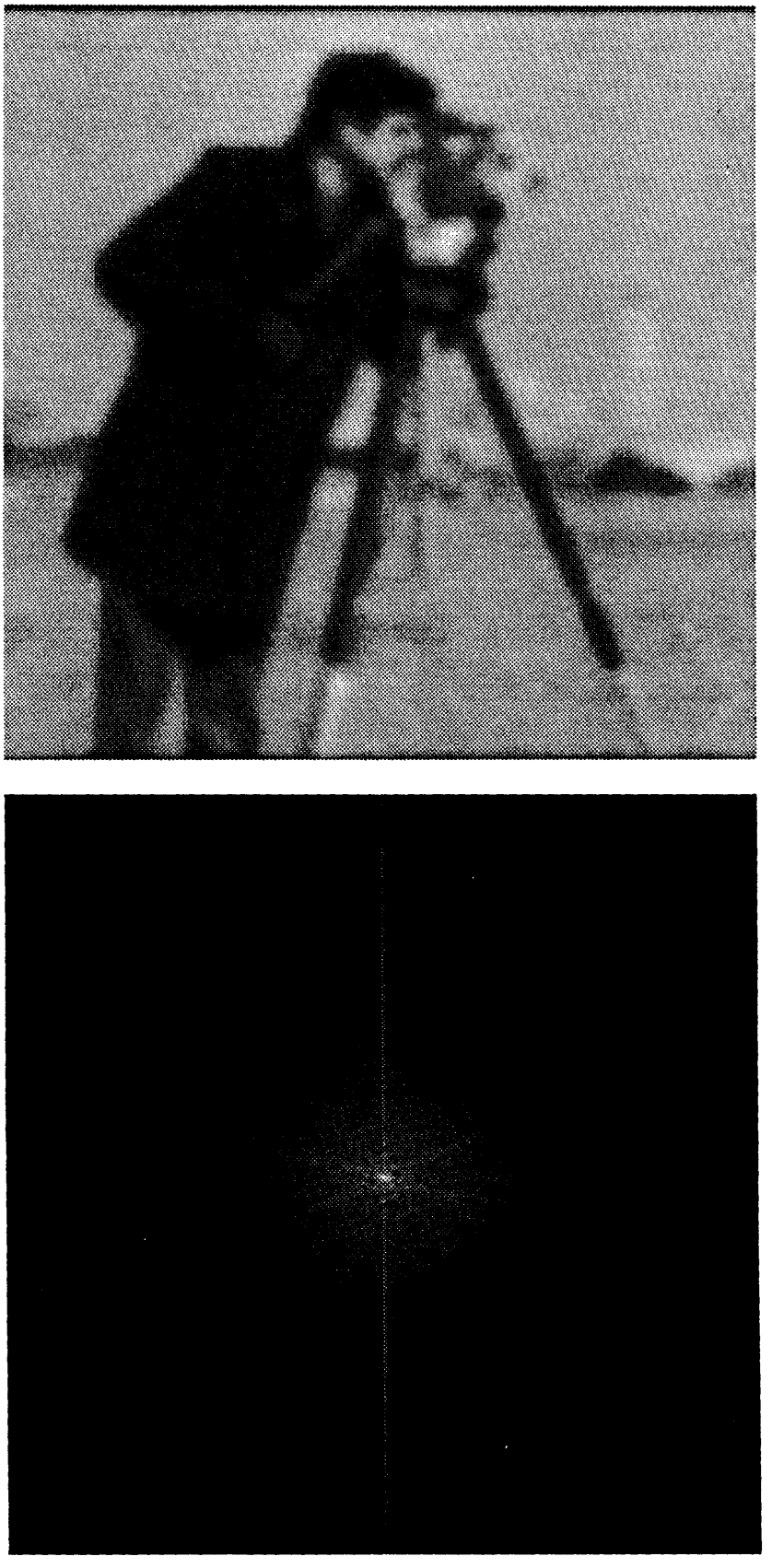


Fig. 2.16(j). Read beam: impulse; write beam: 15H×15V Gaussian

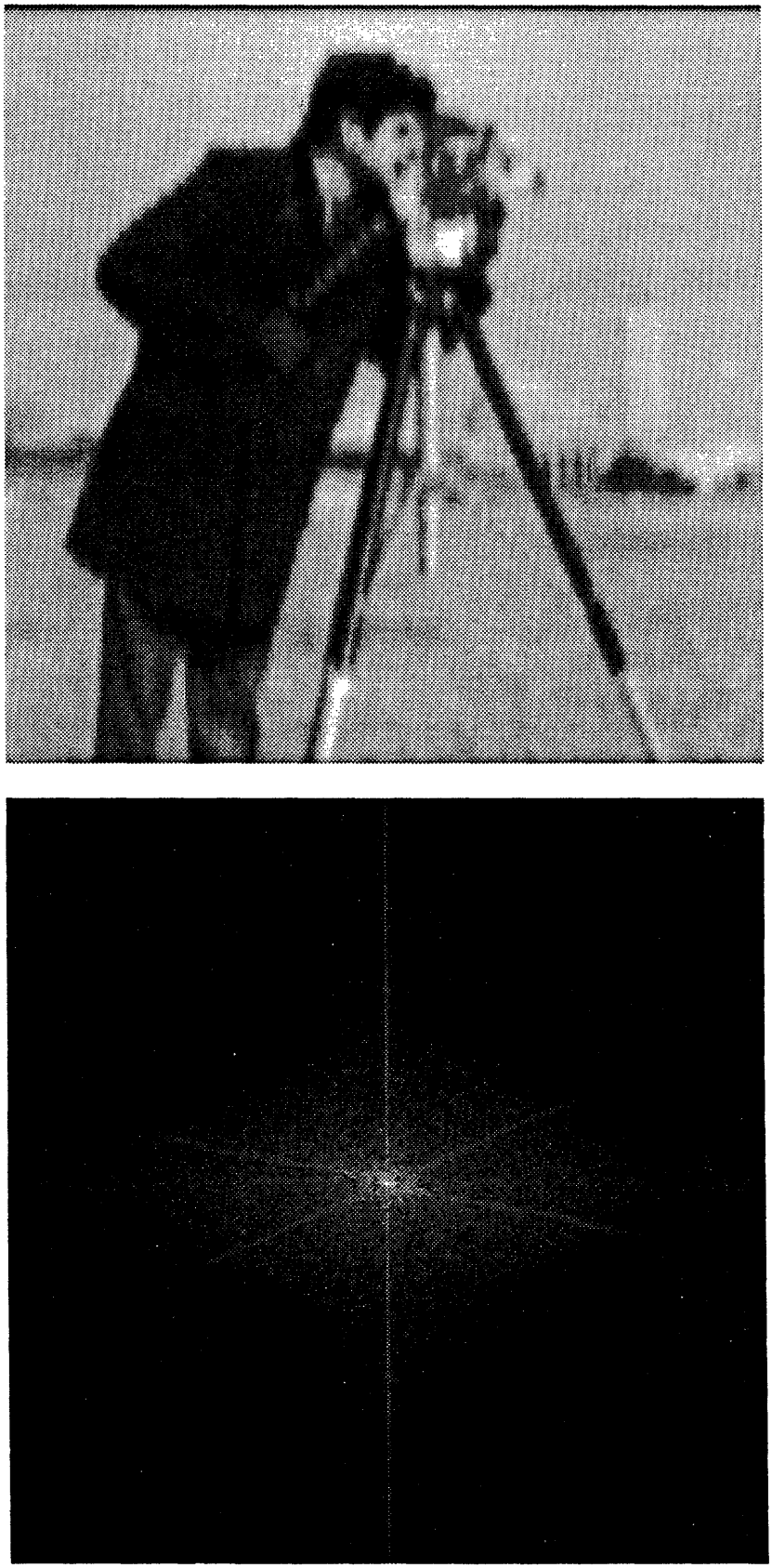


Fig. 2.16(k). Read beam: impulse; write beam: 7H×15V Gaussian

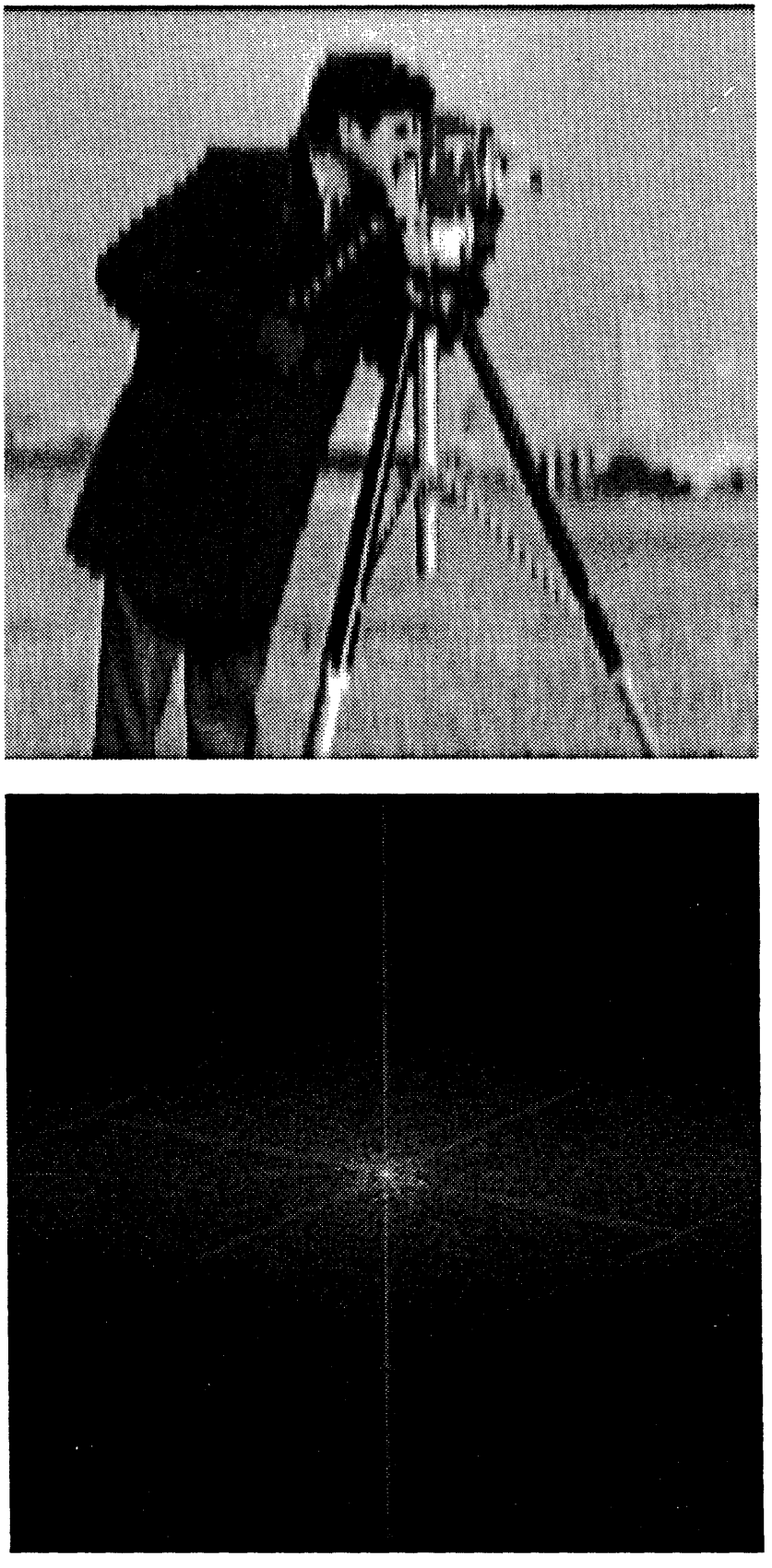


Fig. 2.16(l). Read beam: impulse; write beam: 3H×15V Gaussian

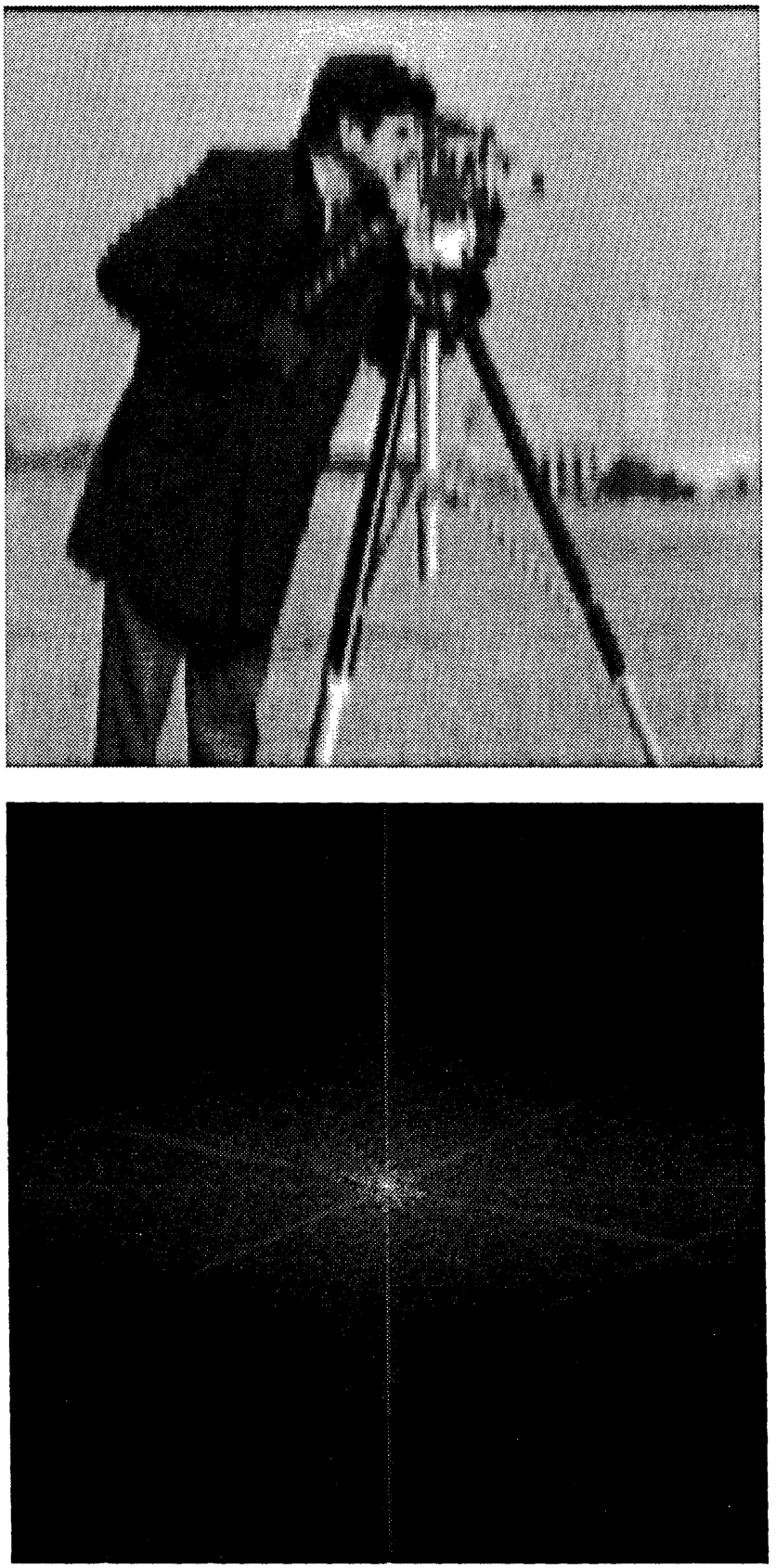


Fig. 2.16(m). Read beam: 3H×3V Gaussian; write beam: 3H×15V Gaussian

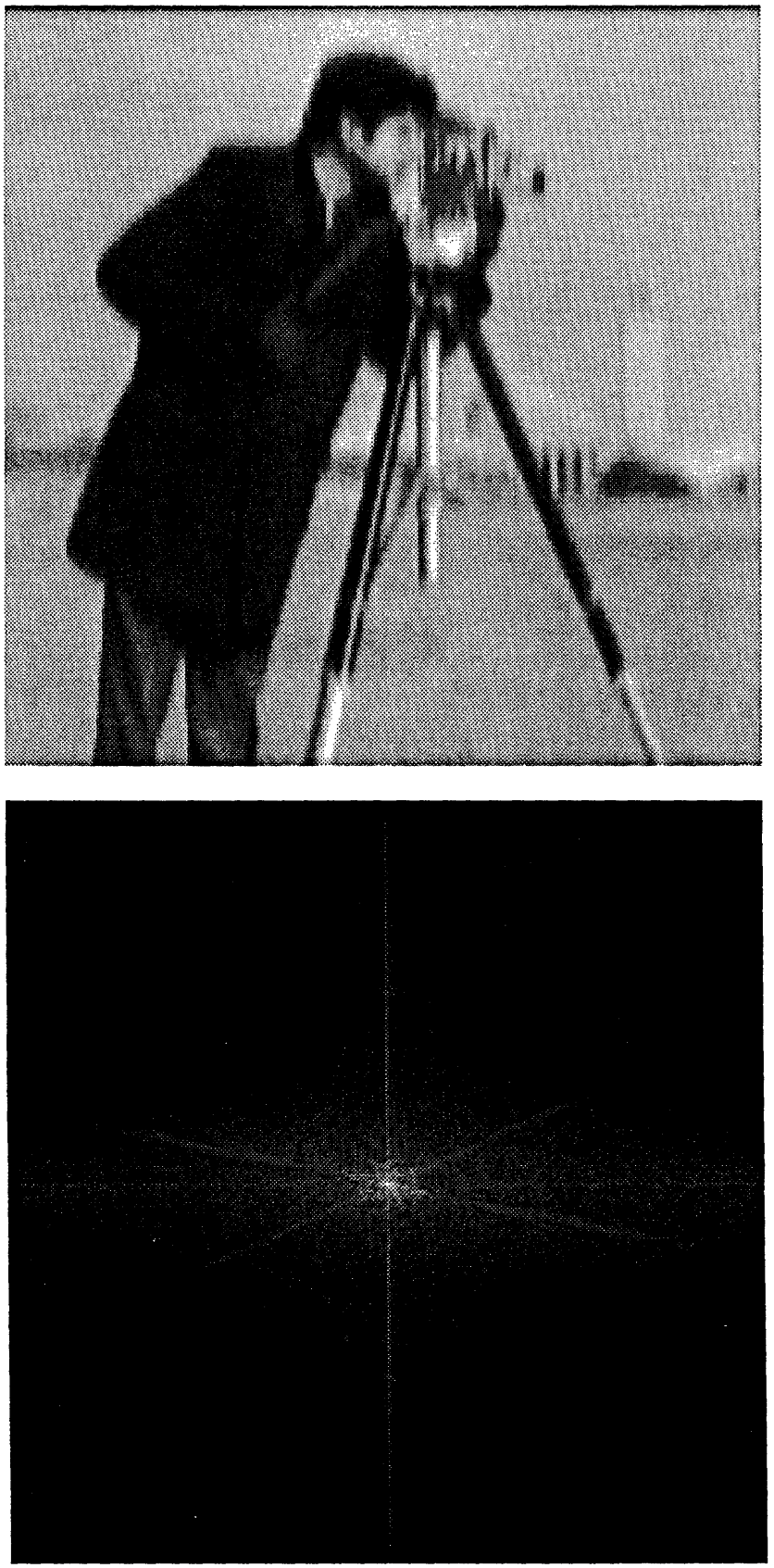


Fig. 2.16(n). Read beam: 3H×9V Gaussian; write beam: 3H×15V Gaussian

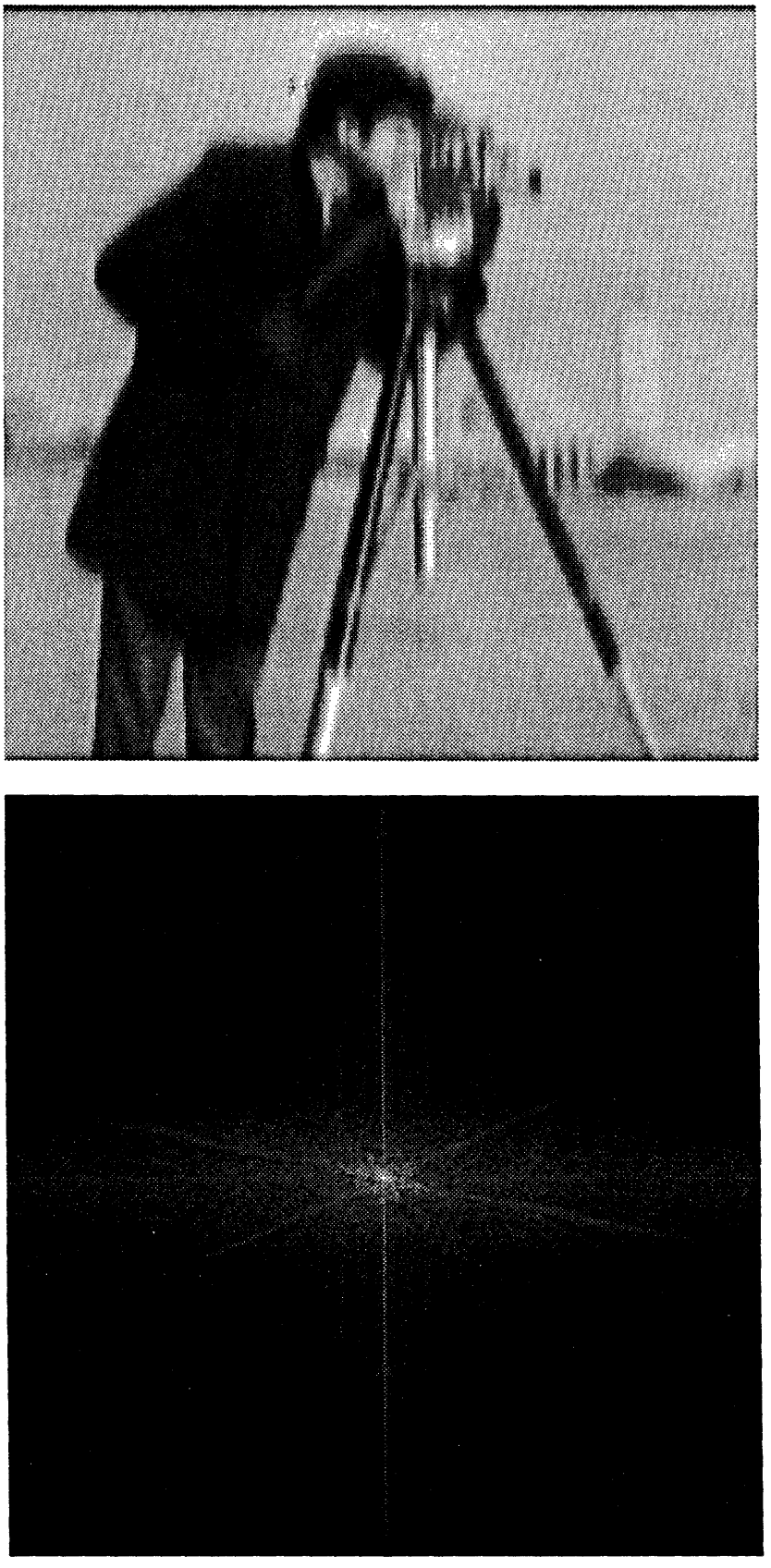


Fig. 2.16(o). Read beam: 3H×15V Gaussian; write beam: 3H×15V Gaussian

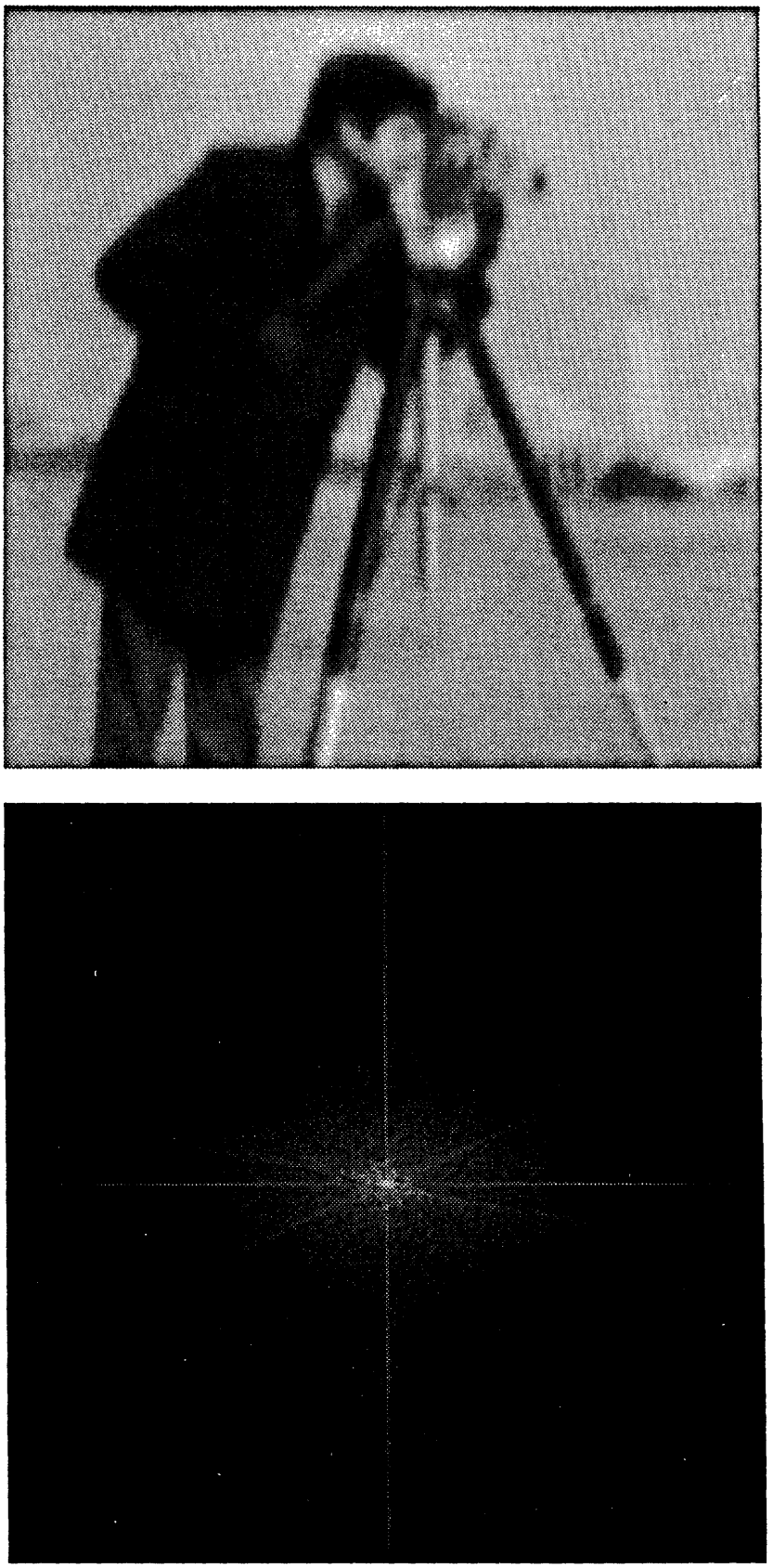


Fig. 2.16(p). Read beam: 9H×9V Gaussian; write beam: 3H×15V Gaussian

elements. The active sensing and display areas are squares ranging in size from 2×2 to 4×4 elements. The resulting set of nine pictures is shown in Fig. 2.17 for the eye image, and in Fig. 2.18 for the test pattern. The halftone printing of these images introduces two artifacts. First, at small values of display areas, the square aperture shape is distorted by the high-frequency halftone screen. Second, the reflectance of the paper limits the maximum reproducible image brightness.

As the percentage of filled area in the display approaches 100%, the pictures become brighter and more natural in appearance. At fill ratios less than 50%, the active areas approach impulses, and the high spatial frequencies of the display grid begin to mask spatial detail. As expected, aliasing decreases as the size of the camera aperture increases. The moire patterns appear both vertically and horizontally because sampling occurs in both dimensions. Depending on the amount of aliasing, the symmetric 2-D moire patterns of the solid-state system may be less objectionable then the vertical 1-D pattern of the tube-type system. Severe aliasing, illustrated by the upper right images in Figs. 2.15 and 2.18, is less objectionable in the solid-state system (Fig. 2.18) because of its symmetry. However, slight aliasing, illustrated by the lower right images in Figs. 2.15 and 2.18, is less objectionable in the tube-type system (Fig. 2.15) because the high-frequency rolloff is more gradual, and jagged edges are suppressed.

2.8.3. Simulation of the HVS Perceptual Response

On close inspection, the line or cell structure of any physical display becomes apparent. However, when viewed at normal distances, the bandpass response of the eye masks these degradations. Aliasing artifacts, consisting of lower spatial frequencies, may still be visible even when sampling structure is not. This is illustrated by the following experiment.

The original luminance pattern is shown in Fig. 2.19(a). When this pattern is scanned with 64 lines by a tube-type camera (σ_r = 0.25 TVL) and displayed on a tube-type display (σ_w = 0.375 TVL), the result is shown in (c). In (d) is shown the perceptual response of (c) when viewed from a distance at which two successive

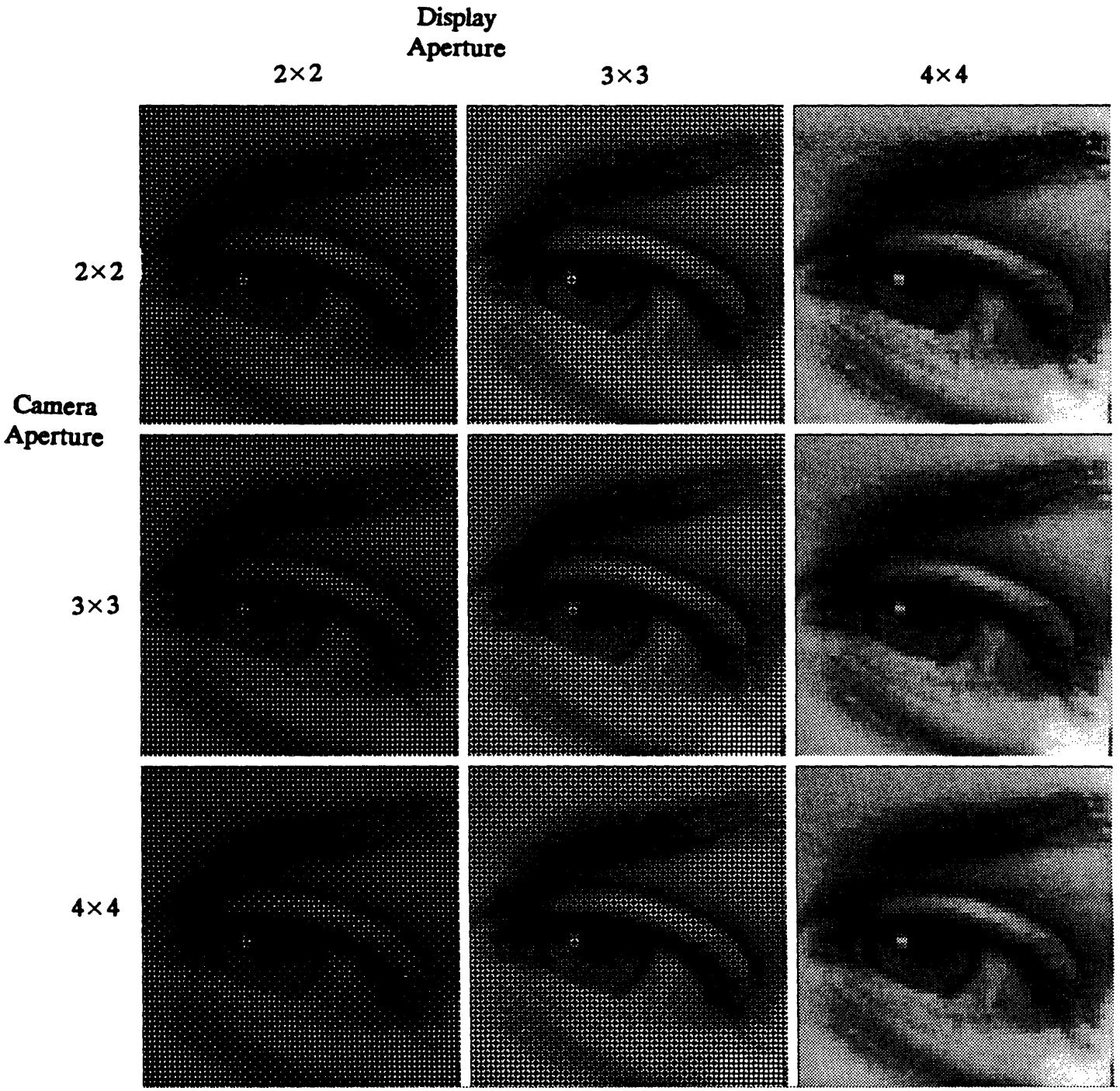


Fig. 2.17. Effect of square aperture sizes on image quality. Natural image.

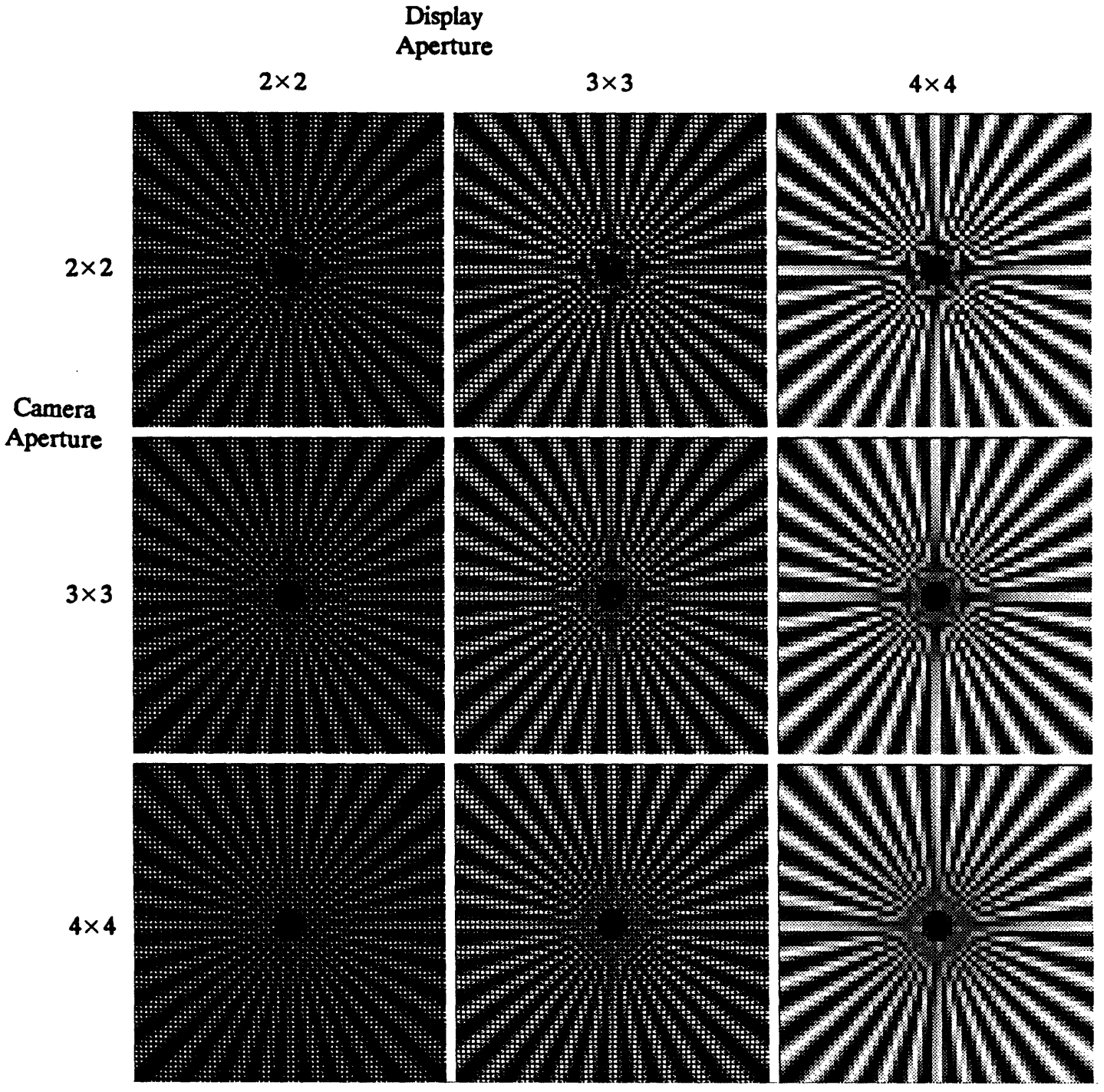


Fig. 2.18. Effect of square aperture sizes on image quality. Test pattern.

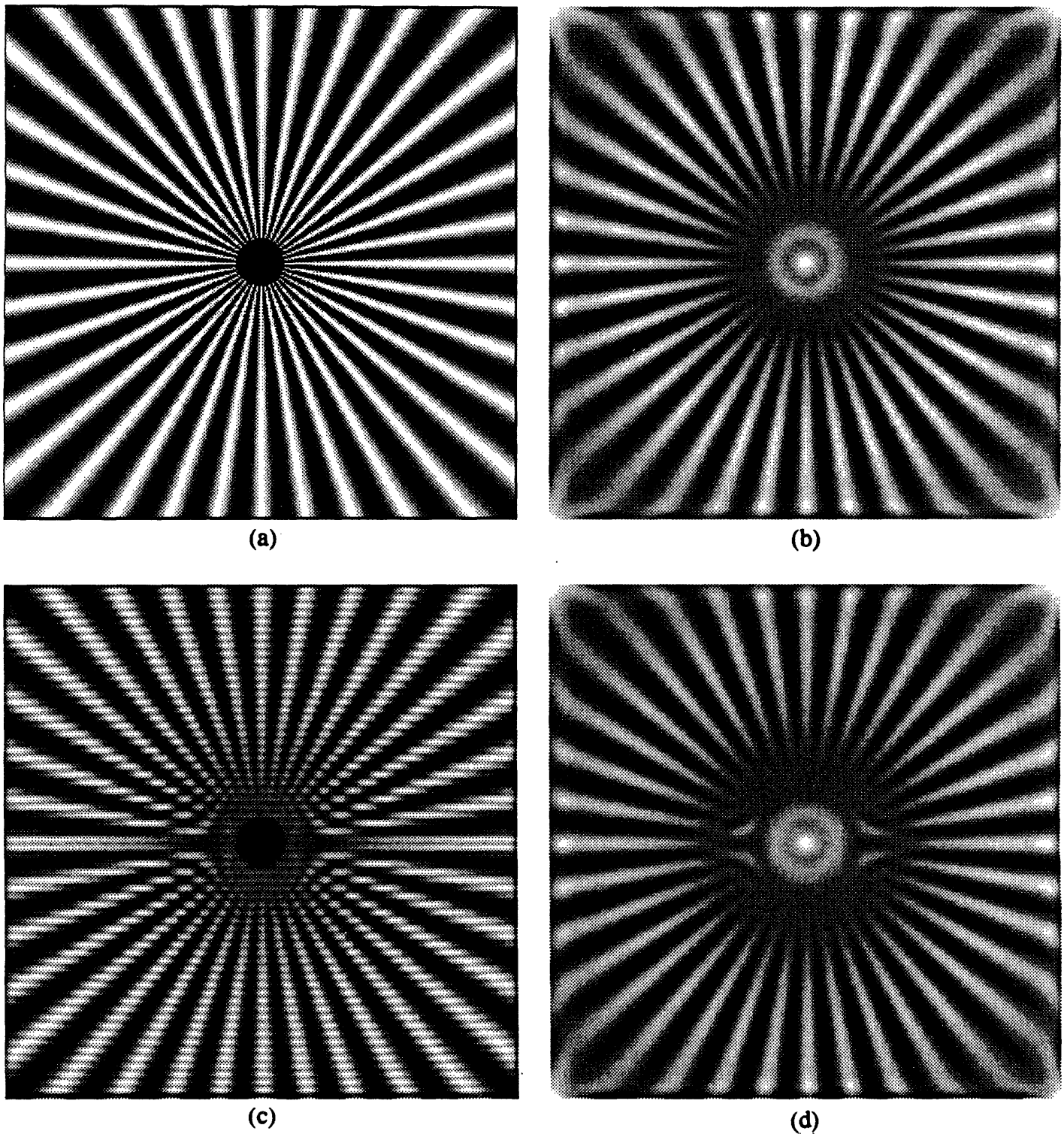


Fig. 2.19. Perception of spatial detail using linear bandpass HVS model. (a) Original image. (b) Perception of (a) at a distance where height of image subtends $\approx 2°$ of arc. (c) Image displayed using 64 scan lines. (d) Perception of (c) at same viewing distance.

scanning lines subtend an angle of approximately 2 minutes of arc. This viewing distance is about 8.4 feet from the page. At the same viewing distance, the perceived response of (a) is shown in (b).

The perceived images are obtained by linearly filtering the originals with a function derived from the difference of two Gaussians that approximates the impulse response shown in Fig. 2.13(b). The bandpass characteristics of the filter produce an interesting phase reversal in the center of the test pattern that is not observed in practice. However, the HVS processing does produce some useful results. First, it predicts a decrease in scan line visibility in the perceived TV image. Second, it shows that aliasing artifacts (moire patterns) are reduced in intensity, but still remain. Finally, it reveals that the perceptual responses of the luminance patterns are more alike than the luminance patterns themselves. Therefore, HVS processing can be used to measure the perceptual match between an original scene and its displayed counterpart.

Numerical HVS models incorporating nonlinearities and threshold levels would undoubtedly yield more accurate results and should be developed as an extension to this work. However, the intent of this experiment is to show how a simple HVS model can be included as a final processing step in a numerical simulation of the TV process.

2.9. Summary

In this chapter, functional modeling was applied to the spatial processes of a monochrome baseband television system. A camera lens was modeled as a linear filter, and common aberrations were modeled by simple space-variant processing. Various charge scanning models were developed. Numerical simulations verified the beam sharpening effect. Linear erasure has the highest sharpening, but the most distortion. It can also be used to simulate the steady-state appearance of images scanned by low beam currents. The effect of beam width on image quality was investigated. It was shown that for natural imagery, the read beam sigma should be approximately 0.2 TVL and the write beam sigma should be about 0.4

TVL for good subjective results. Aliasing and blur were studied in both the spatial and frequency domains. The effect of square uniform aperture sizes on image quality was studied. To reduce aliasing, the camera fill ratio should be greater than 50%; to reduce sampling structure visibility, the display fill ratio should be greater than 90%. Finally, a linear bandpass model for the HVS predicted greatly reduced scan line visibility and slightly reduced aliasing visibility at normal viewing distances.

Chapter 3

A Temporal Model of a Monochrome Baseband TV System

3.0. Introduction

In this chapter, functional modeling is used to investigate the temporal characteristics of a simple monochrome baseband television system. Section 3.1 focuses on the behavior of a single camera element connected to its corresponding display element, assuming a noiseless channel. In section 3.2, the temporal model is extended to include both spatial dimensions. This allows analysis of entire sequences of optical frames.

3.1. A Temporal Model of a Television System

In well-designed television systems, spatial resolution is preserved as much as possible from camera to display. This implies the existence of small areas on the camera target and on the display surface that are insensitive to the degrading effects of neighboring areas. As discussed in Chapter 2, the smallest such area is termed a *mosaic element.* In CCD sensors and matrix displays, the mosaic elements may be grouped into larger units, called *pixels*, which have well defined shapes and areas; furthermore, pixels are highly isolated from one another. In tube-type cameras and CRT's, pixels are not so well defined; interaction between pixels may cause resolution loss or other undesirable artifacts.

In this section, functional modeling is used to study the temporal characteristics of a single camera element connected to its corresponding display element. Starting with incident illumination on the camera element, the temporal processes of light-to-charge conversion, charge readout, gamma correction, and charge-to-light conversion at the display element are modeled mathematically. The temporal

characteristics of the human visual system (HVS) are considered as well. Waveforms associated with these processes are generated from this software model and, where possible, compared to those of real devices.

The block diagram in Fig. 3.1 shows the important temporal processes in a monochrome television system. Ignored in this functional model are gain stages, noise sources, and degradations from neighboring camera and display elements. A time-varying flux of photons, $e_C(t)$, is the incident illumination on the camera pixel, and a flux of photons, $l_D(t)$, is the luminous output of the display pixel. The HVS interprets $l_D(t)$ as a time-varying brightness function, $b_D(t)$. The goal of TV is to create a perceptual match between the perceived image illuminance and the perceived brightness of the display.

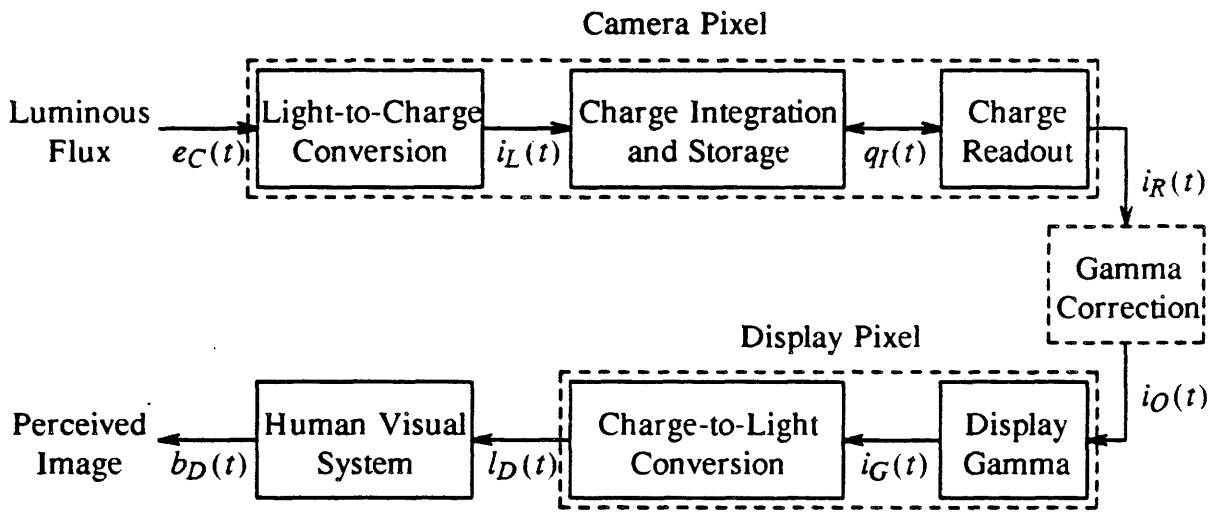


Fig. 3.1. Identifiable temporal processes in a single camera and display element.

3.1.1. Modeling the 1-D Image Illuminance Function

The input to the camera element is the *imaged illuminance,* $e_C(t)$, of a scene before the camera. It is usually sufficient to model the light flux as a purely deterministic waveform, such as a square wave or sine wave; however, a more accurate model would incorporate the Poisson-like statistics of photon flux [70]. Stochastic modeling is necessary at extremely low light levels or within very small time intervals [71, 72]. In most cameras, other noise sources, such as shot noise in camera read beams, are more significant than flux variances. For these reasons, only deterministic flux functions are considered here.

3.1.2. Modeling the Dynamics of a Camera Element

Operating on the light flux, $e_C(t)$, are several identifiable temporal processes. These are shown in Fig. 3.1.

3.1.2.1. Light-to-Charge Conversion.

Light-to-charge conversion is the first photoelectric process; it may be photoemissive or photoconductive in nature. Ignoring the effects of subsequent processes, the photocurrent, $i_P(t)$, generated by the light input is given by $i_P(t) = f_1(e_C(t))$, where $f_1(\cdot)$ is a function whose parameters are determined by the specific imaging device. For instance, the relation

$$i_P(t) = G \cdot e_C(t) \tag{3.1}$$

describes a linear dependence of photocurrent on light input. The parameter G is the *photoconductive gain* of the sensing element. Most semiconductors do not respond in such a linear fashion [73]. The mechanism of electron-hole recombination in the semiconductor may produce nonlinear light-to-charge characteristics. Bimolecular recombination, for instance, predicts a photocurrent proportional to the square root of light intensity [57, 39]. A more general relation that approximates

typical nonlinear recombination effects is

$$i_P(t) = G \cdot e_C(t)^{\gamma_1} + i_d \,, \quad \gamma_1 \leq 1, \tag{3.2}$$

where γ_1 is the *photoconductive gamma,* or *tonal transfer characteristic* of the sensor, and i_d is the *dark current* that flows even when $e_C(t) = 0$. In general, both γ_1 and i_d are functions of other camera parameters, such as V_T, the target voltage. When $\gamma_1 = 1$ and $i_d = 0$, the device becomes linear. In real devices, the photocurrent may saturate at extreme levels of illumination. Linear sensors, such as Plumbicon tubes, are more susceptible to saturation effects than nonlinear sensors, such as vidicons, whose low values of gamma provide built-in highlight compression. Eq. (3.2) can be easily modified to model hard-limiting saturation effects.

An important temporal characteristic that can be modeled at this stage is *photoconductive lag,* which is strongly dependent on the electron and hole mobilities in the semiconductor [74]. Experiments with camera tubes show that changes in image illuminance do not cause an immediate change in the photocurrent of the camera target. The build-up and decay of photocurrent generally lags corresponding changes in the optical illumination. A step change in image illumination causes a nearly exponential change in the photocurrent that can be modeled rather well by a single time constant [75], although in vidicons, the decay lag is more hyperbolic ($\propto 1/t$) than exponential [39, 51]. Mathematically, exponential lag behavior is described by the first-order equation

$$\frac{di_L(t)}{dt} + \frac{i_L(t)}{\tau_L} = k_L \cdot i_P(t), \tag{3.3}$$

where $i_L(t)$ is the lagged photocurrent, τ_L is the lag time constant, k_L is a constant of proportionality, and $i_P(t)$ is given by Eq. (3.2). More complex lag behavior can be realized by letting τ_L vary as a function of light intensity, temperature, or target voltage [57].

3.1.2.2. Charge Integration and Storage.

The charge integration and storage process interacts with the charge readout process to produce the camera output current, $i_R(t)$. Fig. 3.2(a) is a *temporal integration pattern* of a single camera element. It clearly shows the order in which charge integration, storage, and readout occur during each frame period. For most cameras, the *frame period,* T_F, the *integration period,* T_I, the *storage period,* T_S, and the *readout period,* T_R, are fixed. Since T_F is the temporal sampling period, it affects the temporal resolution of the camera. T_I affects luminous sensitivity, SNR, and motion blur; if $T_I \ll T_F$ little motion blur occurs, but fewer photons are sensed, decreasing the SNR.

Theoretically, T_I may assume any value between 0 and T_F; however, most real devices impose constraints on its value. For instance, in most interlaced tube-type cameras, the scanning spot erases the entire target every field, thus $T_I = T_F/2$. In progressive-scan tube-type cameras, $T_I \approx T_F$. Mechanical shutters can be used to reduce the integration time. Only recently have solid-state sensors been manufactured with the capability of electronically adjusting the integration time to almost any fraction of the frame or field period. This flexibility enables a single sensor to operate in different modes. Small values of T_I record motion stroboscopically, and each video frame is free from motion blur. Large values of T_I produce motion blur, but they may be needed to avoid temporal aliasing.

In a camera sensor, the integrated charge waveform, $q_I(t)$, is related to $i_L(t)$ by

$$q_I(t) = \begin{cases} \int_{\tau=NT_F}^{t} i_L(\tau)d\tau + q_R(NT_F), & NT_F \le t \le NT_F + T_I \\ q_I(NT_F + T_I), & NT_F + T_I \le t < (N+1)T_F \end{cases}, \tag{3.4}$$

where N = integer$[t/T_F]$ is the frame number and $q_R(NT_F)$ is the leftover charge from the previous integration period. Eq. (3.4) states that within each frame time,

charge is integrated for T_I; this integrated charge is held constant until the end of the frame period, when most or all of it is instantaneously read out. In good image sensors, the readout efficiency is nearly 100%, implying that little or no charge remains.

3.1.2.3. Charge Readout.

Charge readout is the final photoelectric camera process. It is unique in that the readout mechanism is practically instantaneous [39]: it samples and modifies the integrated charge waveform during a very short readout interval, $T_R \ll T_F$. For various physical reasons, not all of the available charge is removed at each scan. In tube-type cameras, failure to erase all of the integrated charge during each scan leads to a second type of lag called *beam discharge lag*. The leftover charge contaminates successive frames, resulting in long "tails" that follow moving highlights in the image. Some of the more advanced camera tubes solve this problem by dynamic beam control [68], anti-comet-tail circuitry [75, 76], or through novel gun construction [34, 62].

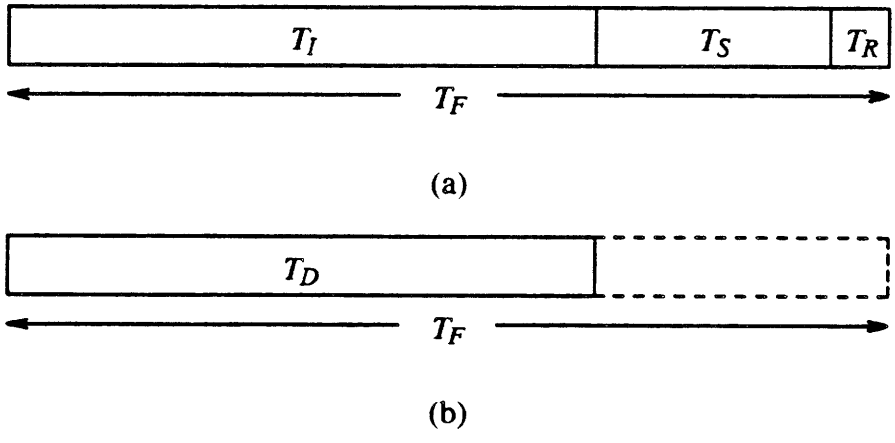


Fig. 3.2. (a) Temporal integration pattern. (b) Temporal interpolation pattern.

As in Chapter 2, two different destructive readout functions can be formulated, corresponding to two different models of the readout process.

Linear Erasure Model. In the linear model, the electron beam is a current source that neutralizes all of the available charge, as long as the beam current is sufficiently high. Any excess beam electrons are turned back and collected by the anode. Thus, the leftover charge at each location is equal to the original charge minus the charge deposited by the beam; if this difference is negative, the leftover charge is zero. Functionally, this can be described as follows:

$$i_R(t) = \begin{cases} q_B/T_R\ , & t = NT_F\ ,\ q_B < q_I(t) \\ q_I(t)/T_R\ , & t = NT_F\ ,\ q_B \geq q_I(t)\ , \\ 0\ , & \text{else} \end{cases} \tag{3.5}$$

$$q_R(NT_F) = \begin{cases} q_I(NT_F) - q_B\ , & q_B < q_I(NT_F) \\ 0\ , & q_B \geq q_I(NT_F) \end{cases}, \tag{3.6}$$

where q_B is the amount of charge deposited by the electron beam during the readout interval, T_R, and q_R is the amount of charge remaining on the element after readout. This model has two interesting features. First, after a light source is turned off, this model predicts a linear decay of remaining charge and a constant-valued readout signal. Second, for a constant source of illumination, if the charge deposited by the beam is sufficiently small, the charge dynamics will saturate before reaching steady state. In other words, certain conditions cause instabilities in the linear model. These observations will be verified in a later section.

Exponential Erasure Model. In the exponential model [60], the electron beam reads out a *fraction* of the available charge during each readout interval. This model takes into account the RC characteristics of common camera targets as well as the energy spread in the electron beam as it passes over a charged element. The sole parameter for this model is the *readout efficiency,* $\eta_R < 1.0$, which is defined as

the ratio of charge removed to charge available at the target [38]. The functional relation between $q_I(t)$ and $i_R(t)$ for this model is

$$i_R(t) = \begin{cases} \eta_R \cdot q_I(t)/T_R , & t = NT_F \\ 0, & \text{else} \end{cases} , \tag{3.7}$$

$$q_R(NT_F) = q_I(NT_F)\cdot(1-\eta_R) , \tag{3.8}$$

where q_R is the charge remaining on the element after readout. In general, η_R depends on the value of $q_I(t)$ at the instant of readout. However, this dependence is suppressed during the steady state, and η_R approaches a constant value [38]. The exponential model has two interesting features. First, when η_R is constant, this model produces an exponential build-up and decay of remaining charge and readout signal. This is commonly observed in many tube-type cameras. Second, for a constant source of illumination, the charge dynamics will always reach a steady state, independent of beam current. In other words, the exponential model is stable. These observations will be verified in a later section.

A more accurate model reflects the dependence of η_R on $q_I(t)$. Because of complex properties of the electron beam and camera target, the readout efficiency typically decreases with the amount of available charge. Measurements reveal a nearly linear region [38] described by the relation

$$\eta_R(q_I) = \begin{cases} \dfrac{(\eta_{R,MAX}-\eta_{R,MIN})}{q_{I,MAX}} \cdot q_I , & 0 \le q_I \le q_{I,MAX} \\ \eta_{R,MAX} , & q_I > q_{I,MAX} \end{cases} \tag{3.9}$$

where $\eta_{R,MAX}$ is the readout efficiency at normal-to-high levels of available charge, and $\eta_{R,MIN}$ is the efficiency at levels of charge approaching zero. Incorporation of

this dependence allows the simulation of an important observed characteristic of many modern camera tubes: beam lag is worse at lower levels of illumination. Because of this, some tube manufacturers add a small amount of "bias light" to the rear surface of the photoconductor to artificially raise the charge level, thereby increasing the readout efficiency of the beam. A bias current is subtracted from the readout signal to preserve the DC level. In a later section, signal-dependent lag and lag improvement with bias lighting is simulated.

3.1.2.4. Solid-State Image Sensors

Since modern solid-state devices have nearly 100% charge readout and 100% charge transfer efficiency [77], the exponential model can characterize CCD operation by setting $\eta_R = 1.0$.

3.1.2.5. Gamma Correction.

Gamma correction is an electronic modification applied to the camera output. Its purpose is to compensate for the inherent high contrast of most picture tubes. In some cameras, such as vidicons, gamma correction is not needed because the camera's photoconductive gamma is already matched to that of the picture tube. The effect of gamma correction on the output current, $i_O(t)$, is

$$i_O(t) = i_R(t)^{\gamma_2} . \tag{3.10}$$

The overall gamma of the camera is given by $\gamma_c = \gamma_1 + \gamma_2$.

3.1.3. Modeling the Dynamics of a Display Element

3.1.3.1. Display Gamma.

Denoted γ_d, the display gamma may be an intrinsic property of the device (e.g., CRT's), or it may be artificially introduced. If the charge-to-light conversion is linear, then the entire television system will be linear if $\gamma_d = 1/\gamma_c$. The gamma-modified signal, $i_G(t)$, is the input to the charge-to-light converter and is given by

$$i_G(t) = i_O(t)^{\gamma_d} . \tag{3.11}$$

3.1.3.2. Charge-to-Light Conversion.

Charge-to-light conversion is the final photoelectric process. It is performed in CRT's by electron bombardment of phosphor particles, and in flat-panel displays by gas discharge, LED's, or light valves. This process is almost linear in the region between cutoff and saturation, and may be modeled by

$$l_D(t) = C_D \cdot i_G(t) * h_D(t) + L_D , \tag{3.12}$$

where $l_D(t)$ is the photon flux emanating from the display pixel, $i_G(t)$ is the modulated impulse train of sampled light values, $h_D(t)$ is the temporal impulse response of the charge-to-light system, C_D is a contrast scaling factor, and L_D is a luminance offset. Fig. 3.2(b) shows the *temporal interpolation pattern* associated with a single display pixel. As shown in Fig. 3.3, for CRT's, $h_D(t)$ is nearly exponential [78]:

$$h_D(t) = h_0 \cdot e^{-t/\tau_D} \cdot u(t) , \tag{3.13}$$

where h_0 a scaling factor, τ_D is the time constant associated with phosphor decay, and $u(t)$ is the unit step function. For solid-state display devices, $h_D(t)$ may be

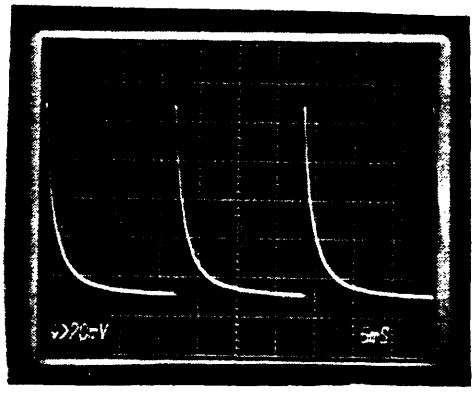


Fig. 3.3 Phosphor decay curve.

more nearly rectangular:

$$h_D(t) = \begin{cases} h_0 , & 0 < t < T_F \\ 0 , & \text{else} \end{cases} . \tag{3.14}$$

3.1.4. Temporal Models of the Human Visual System

The final temporal process is perceptual in nature. It occurs in the human visual system, which is not fully understood. It is known, however, that the gross temporal frequency response is bandpass in nature [79], as shown in Fig. 3.4 along with the temporal impulse response [80]. Most temporal models incorporate non-linearities, but there is no general agreement on their functional form or at what processing stage they occur. For some applications, these details are not important. One simple temporal model of the HVS is a point-wise nonlinearity followed by a

linear filter [81]:

$$b_n(t) = f_n(l_D(t)) , \tag{3.15}$$

$$b_D(t) = h_{HVS}(t) * b_n(t) , \tag{3.16}$$

where $f_n(\cdot)$ is a suitable nonlinearity (*e.g.*, a log function), $b_n(t)$ is the response to the nonlinearity, and $h_{HVS}(t)$ is the temporal impulse response of the human visual system. Kelly and Savoie [80] proposed a filter-log-threshold model to explain flicker and other transient responses.

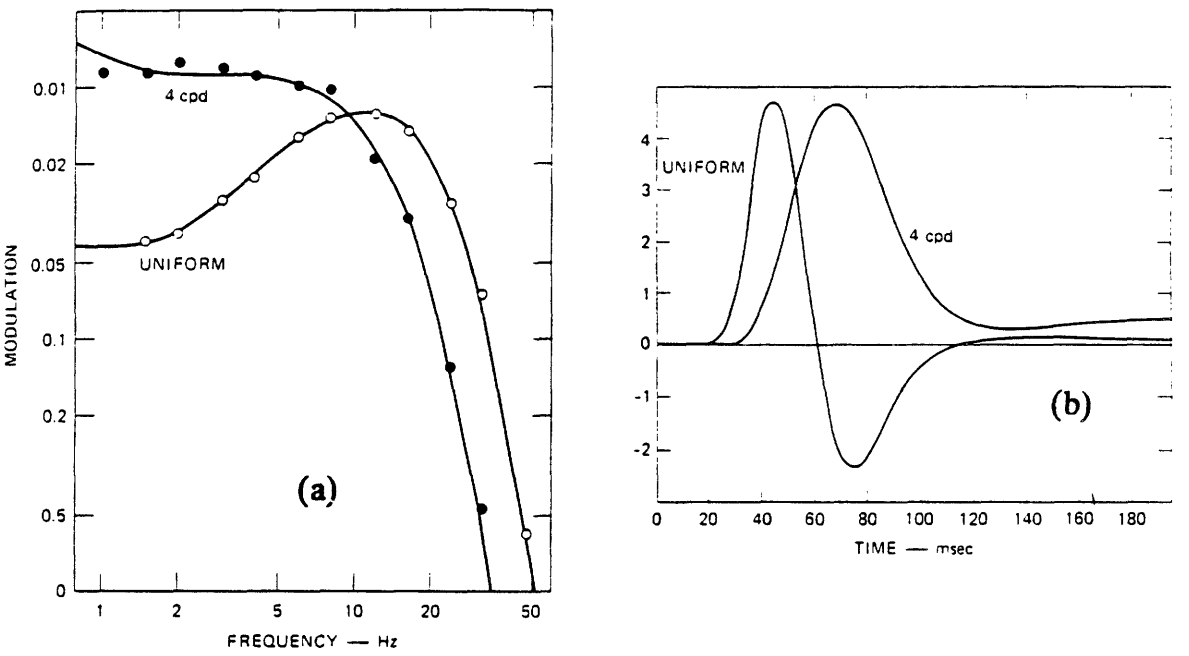


Fig. 3.4. (a) HVS temporal frequency response. (b) HVS temporal impulse response.

3.1.5. Experiments

The program *pel* models the temporal characteristics of a single camera element connected to its corresponding display element, assuming a noiseless channel. The basic unit of time is T_F, the frame period. The program simulates the temporal integration pattern shown in Fig. 3.2(a). The integration period, T_I, is a parameter; the readout interval, T_R, is assumed to be instantaneous. The following experiments illustrate important system waveforms associated with various camera/display configurations. Each simulation spans 30 or more frames (30 frames $\approx$ 1 sec. for NTSC systems) with 50 time samples per frame.

Response to a pulse of light. In Figs. 3.5(a) to (g), the input is a pulse of light lasting 10 frames. Fig. 3.5(a) shows a camera element with no lag of any kind: the light-to-charge process is instantaneous and readout is 100% efficient. Here, $T_I = T_F$. The display element is a phosphor dot with a decay time constant of $0.25T_F$. In Fig 3.5(b), the integration period is shortened: $T_I = 0.25T_F$. The readout signal is decreased in amplitude because less charge has been integrated. In operation, this is similar to the old image dissector tubes, in which there was no charge integration. Fig. 3.5(c) shows a camera element with excessive photoconductive lag. The light-to-charge process has a time constant $\tau_L = T_F$. The readout mechanism, however, is still 100% efficient, so the only contribution to camera lag is in the photoconductor. The display parameters are the same as in (a). Fig. 3.5(d) shows a camera element having a small amount of beam discharge lag but no photoconductive lag. The readout efficiency is 90%. Fig. 3.5(e) is the same as (d), except that the readout efficiency is only 50%. Notice that the readout signals in Figs. 3.5(e) and (c) are quite similar in shape and in value, but they are produced by *completely different lag mechanisms* in the camera element. In Fig. 3.5(f), the camera exhibits both photoconductive lag and beam discharge lag. The former is more significant; this is typical of a vidicon camera tube. The camera of Fig. 3.5(g) also exhibits both types of lag; however, beam discharge lag is more significant. This is typical of low-lag camera tubes such as Plumbicons [82] and Saticons [34, 83].

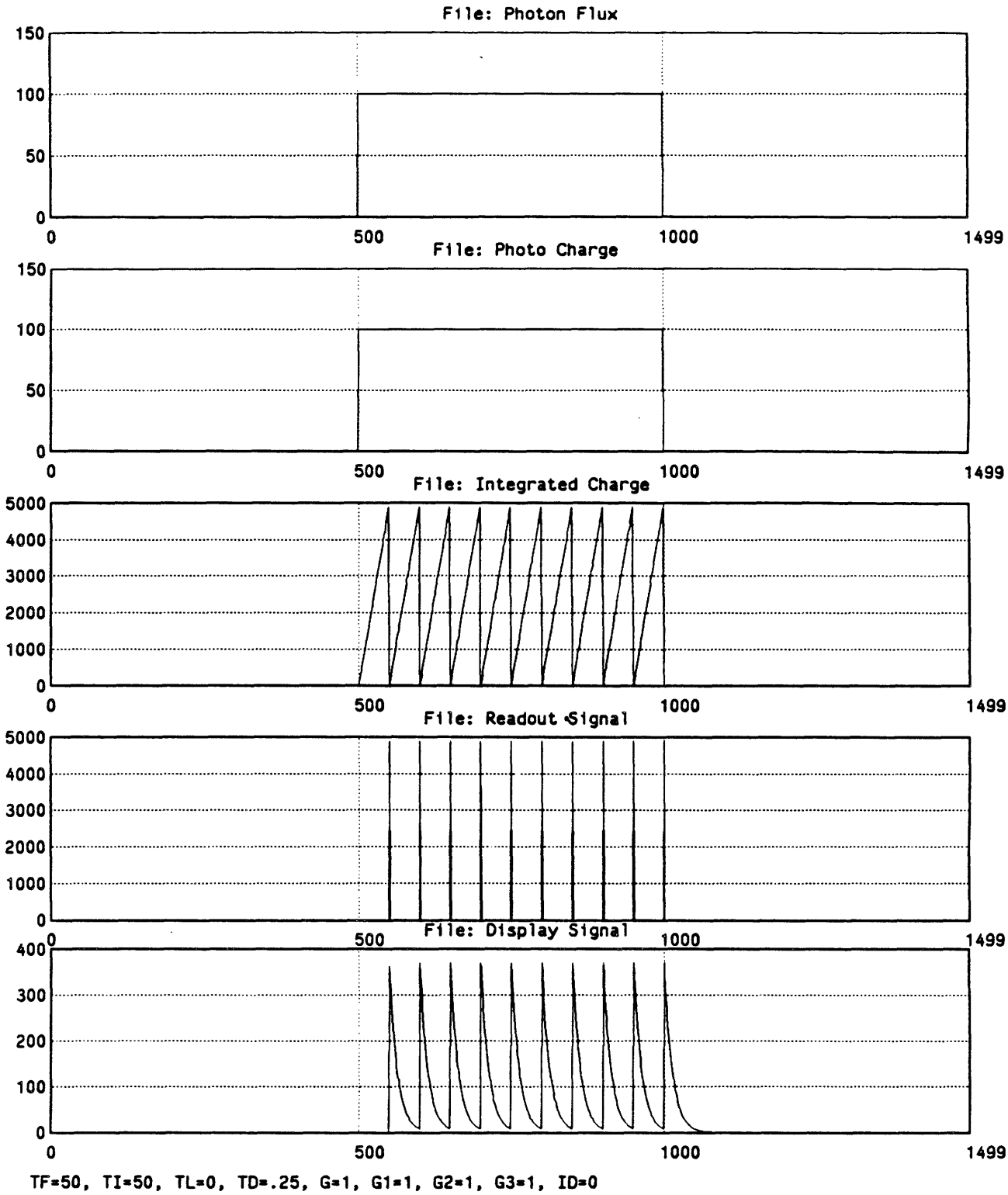


Fig. 3.5(a). Linear TV system with no lag and with $T_I = T_F$.

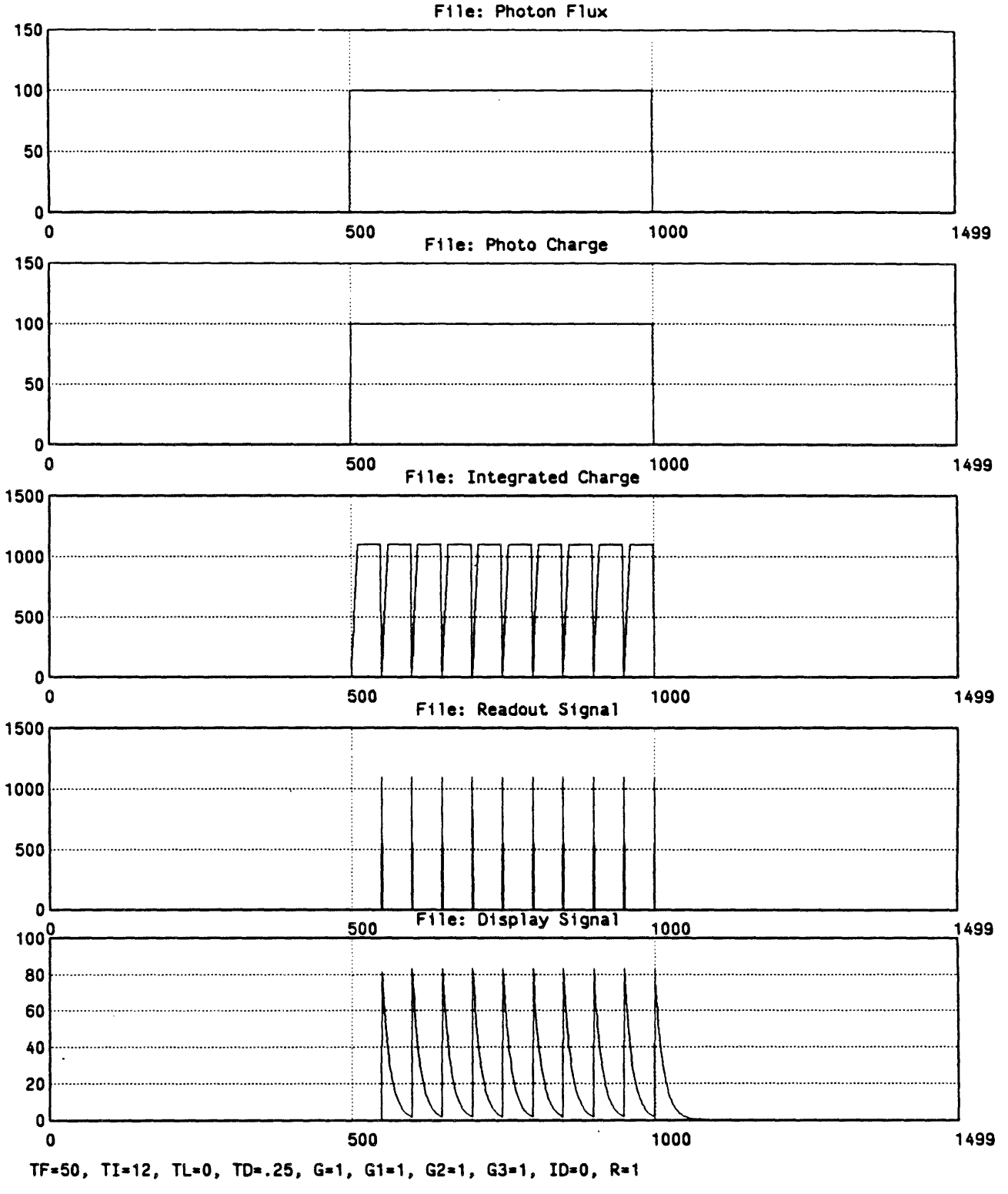


Fig. 3.5(b). Linear TV system with no lag and with $T_I = 0.25 \cdot T_F$.

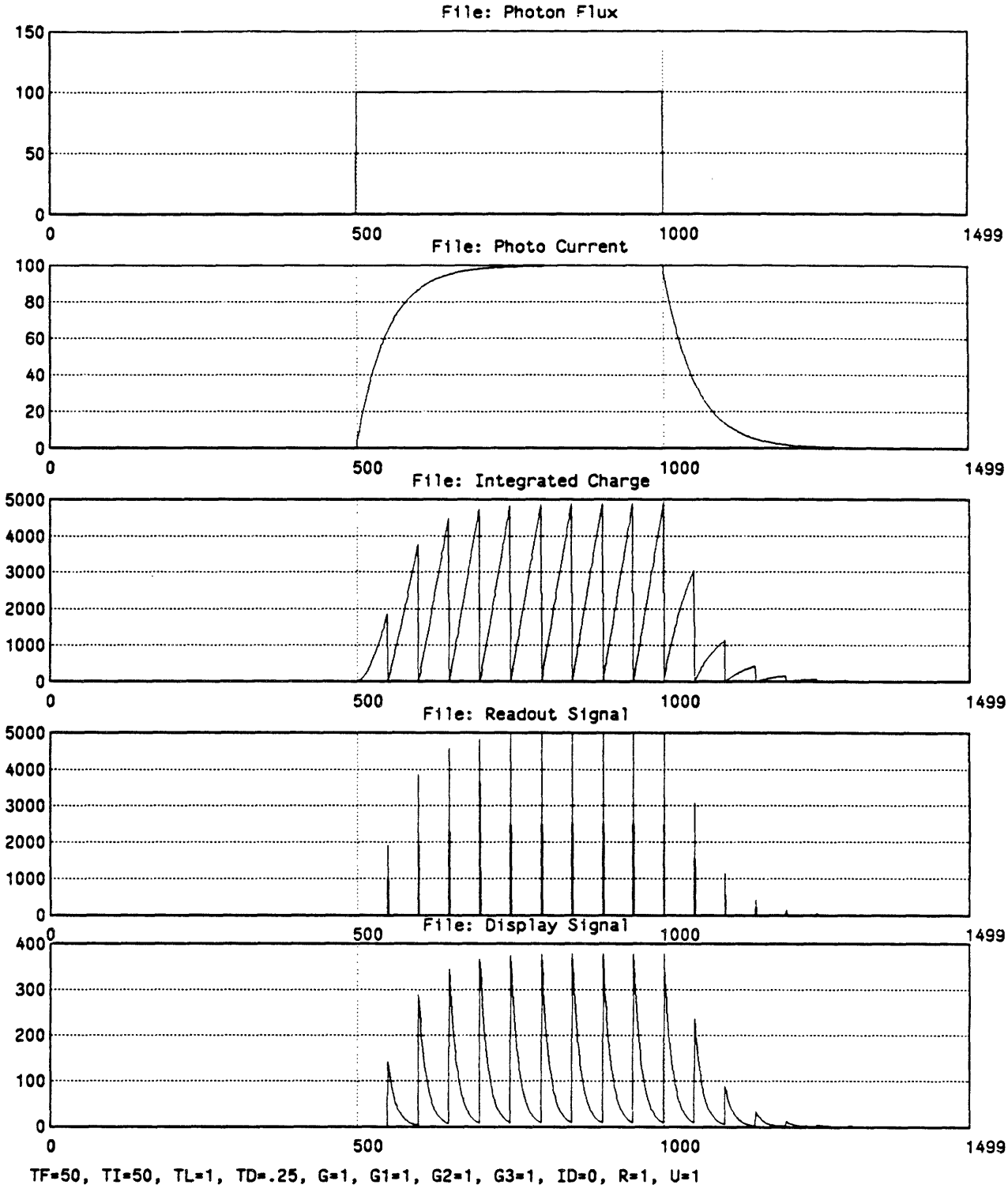


Fig. 3.5(c). Linear TV system with photoconductive lag.

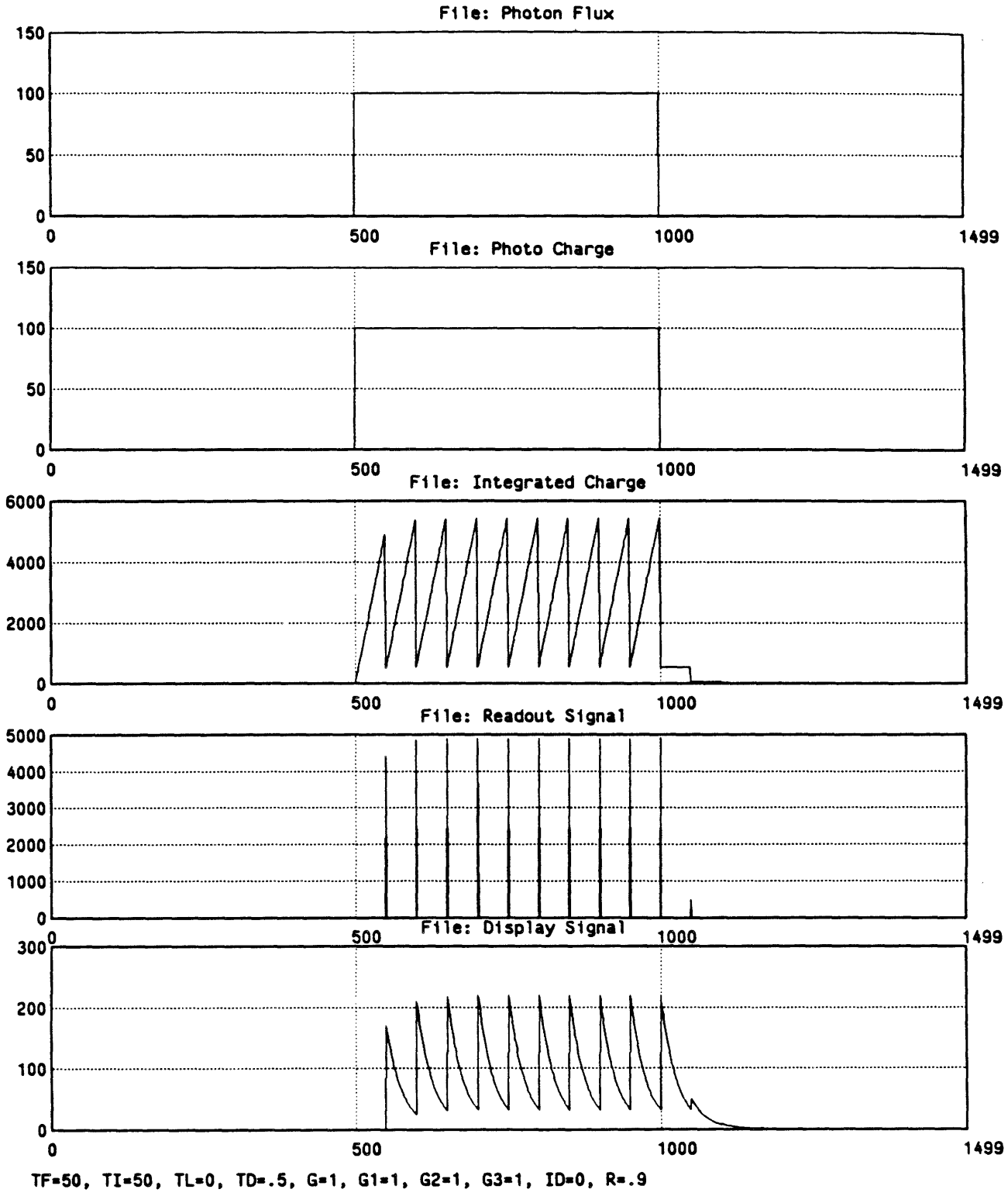


Fig. 3.5(d). Linear TV system with beam discharge lag and 90% readout efficiency.

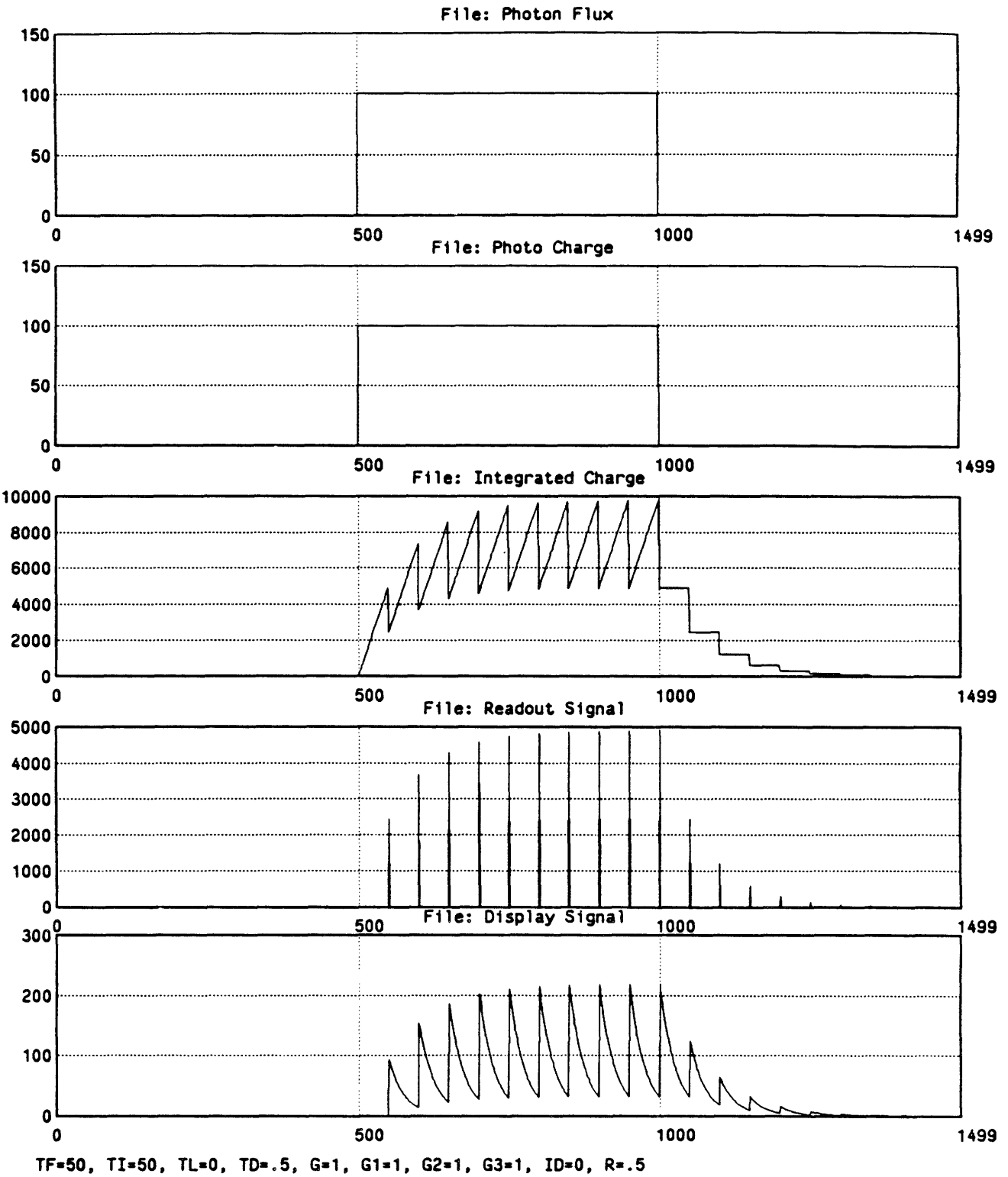


Fig. 3.5(e). **Linear TV system with beam discharge lag and 50% readout efficiency.**

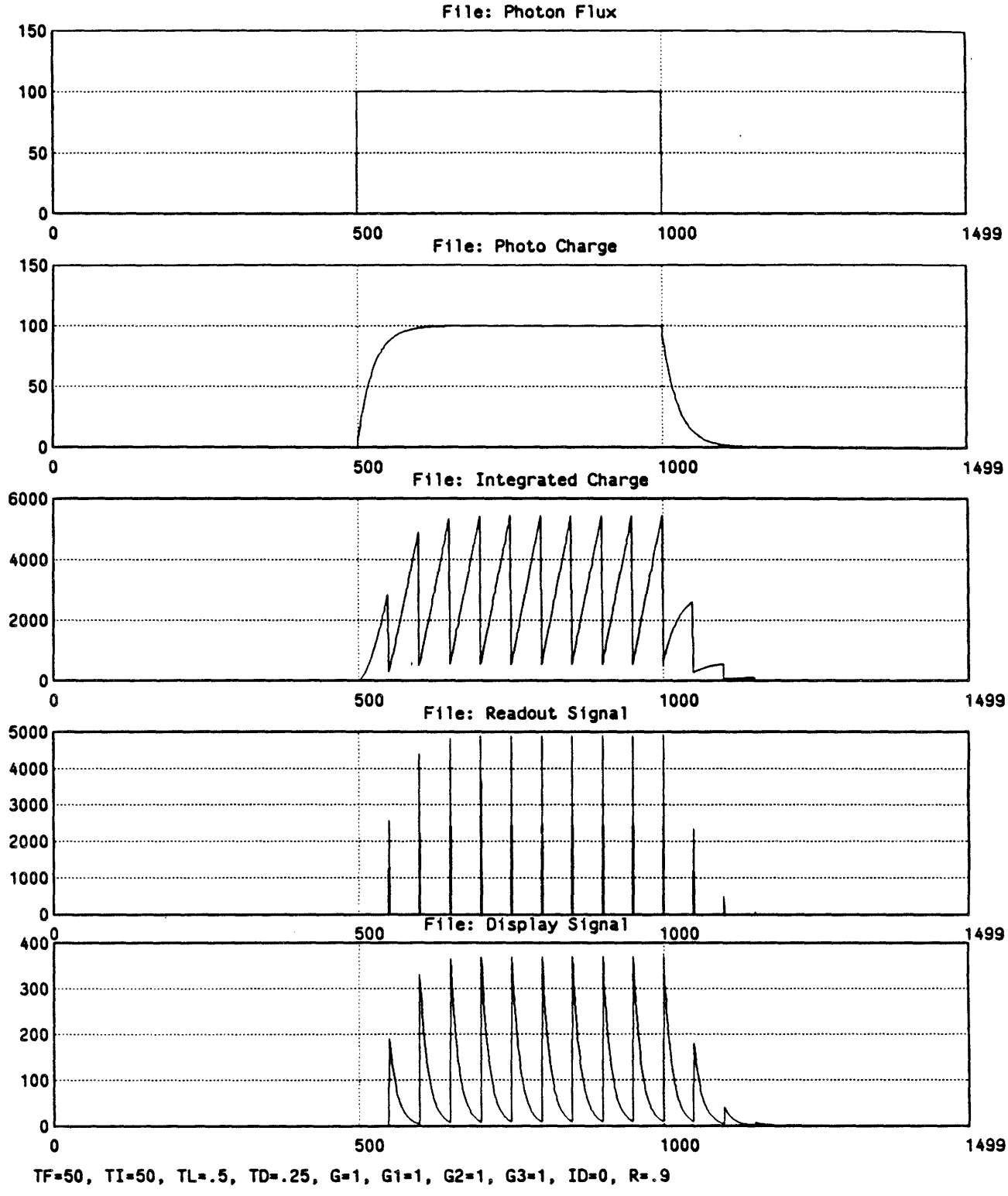


Fig. 3.5(f). Vidicon-type TV system with photoconductive lag and beam discharge lag.

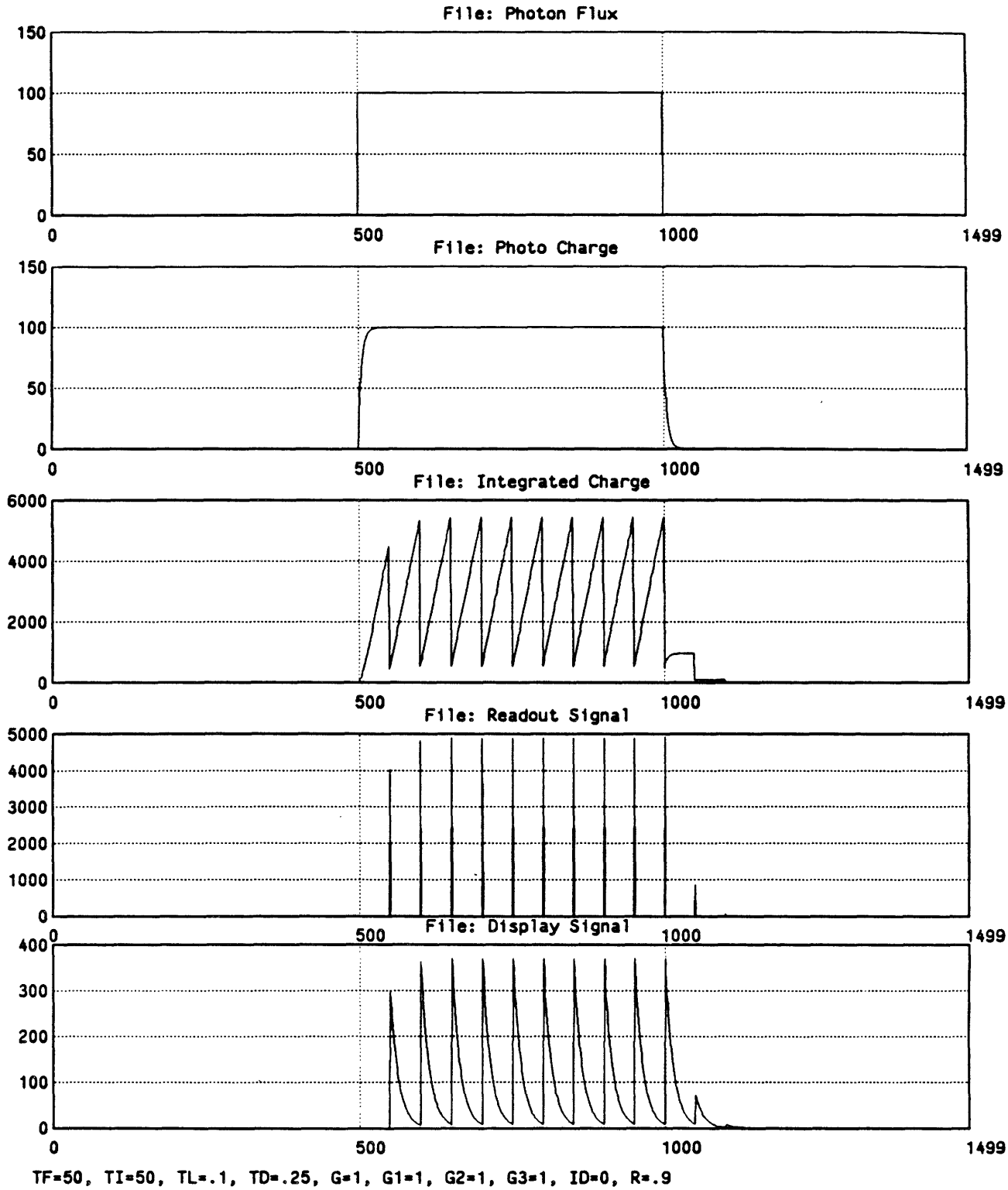


Fig. 3.5(g). Plumbicon-type TV system with photoconductive lag and beam discharge lag.

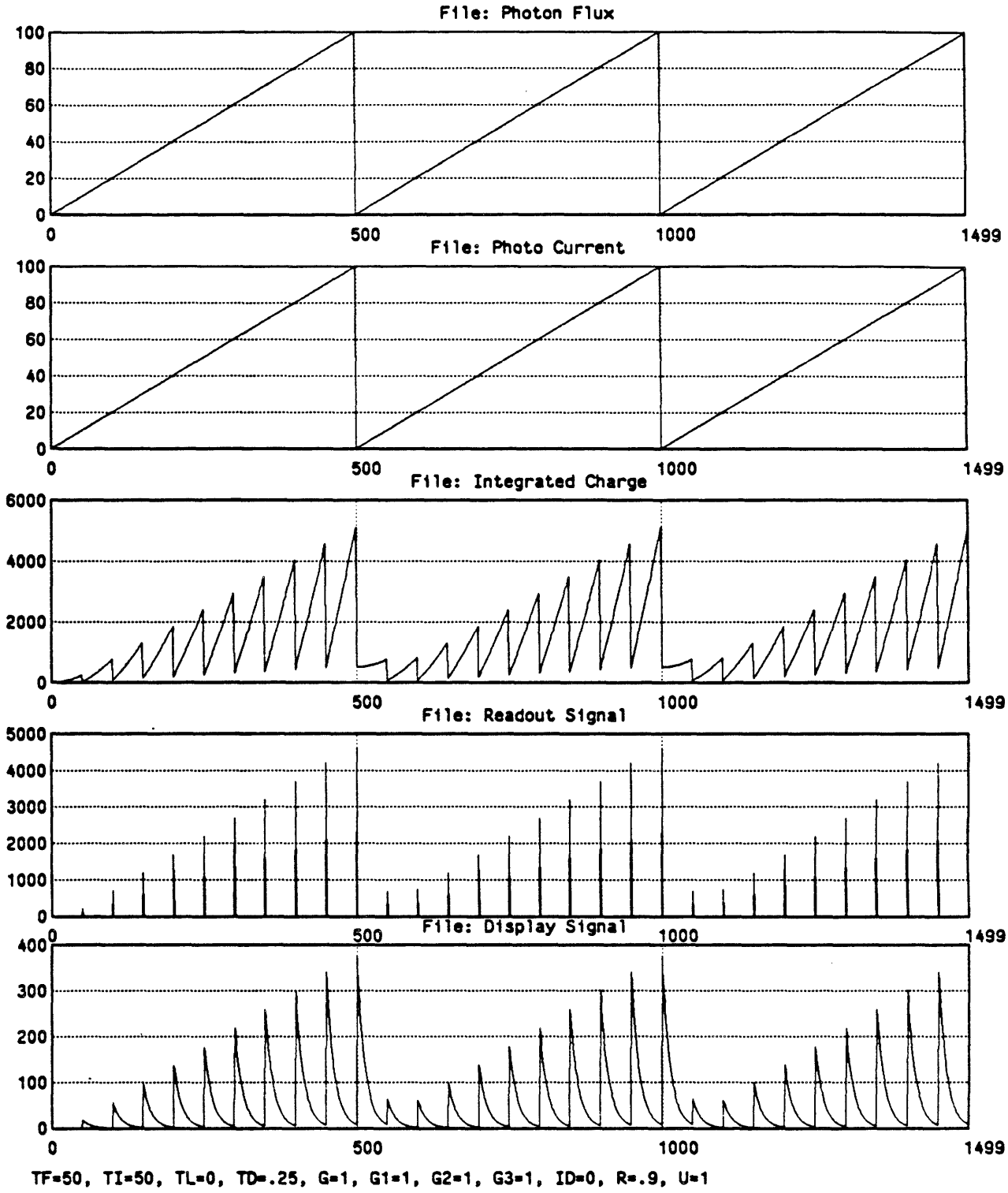


Fig. 3.5(h). Linear TV system with beam discharge lag and 90% readout efficiency.

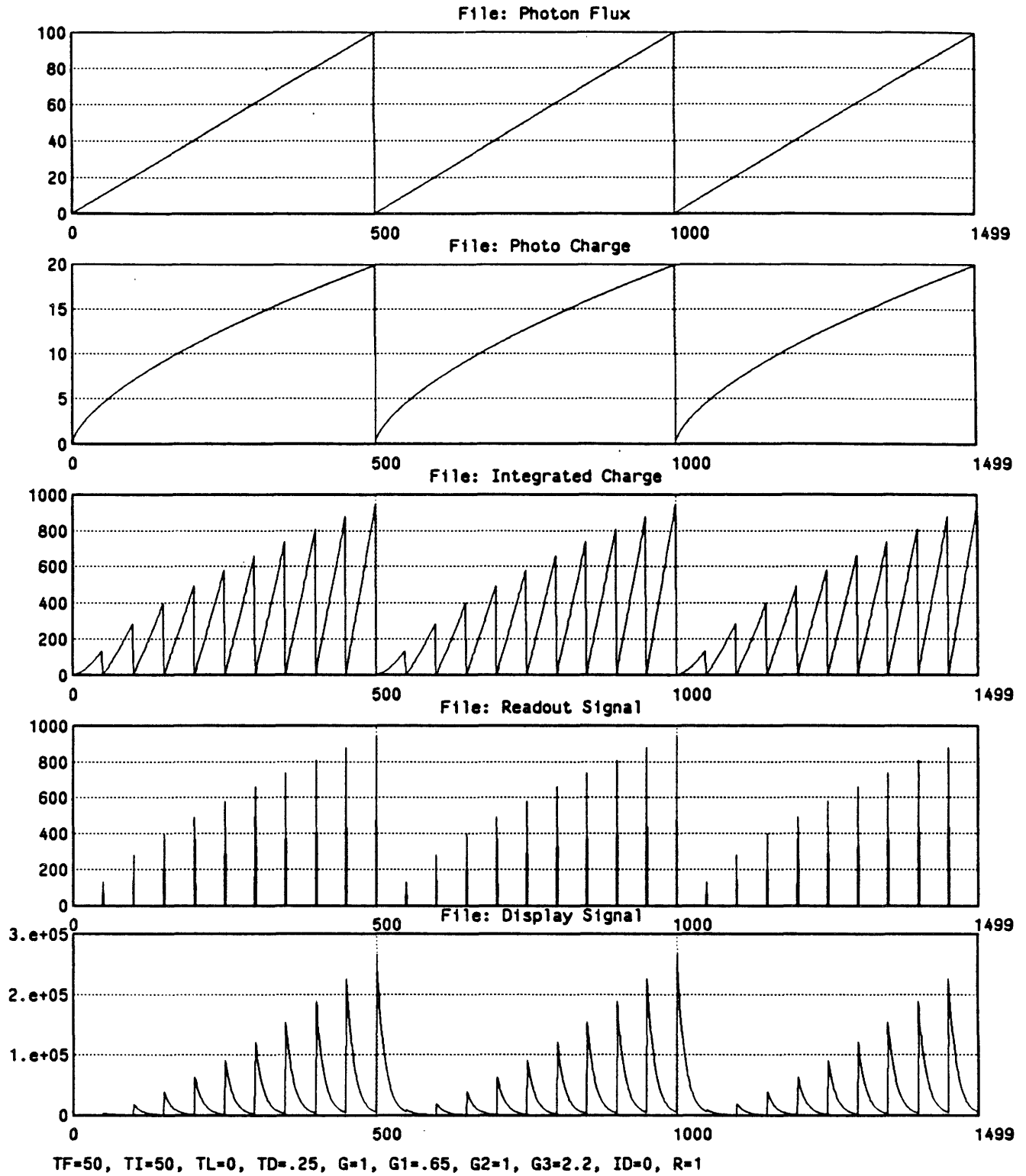


Fig. 3.5(i). Lag-free TV system with $\gamma_1 = 0.65$ and $\gamma_D = 2.2$.

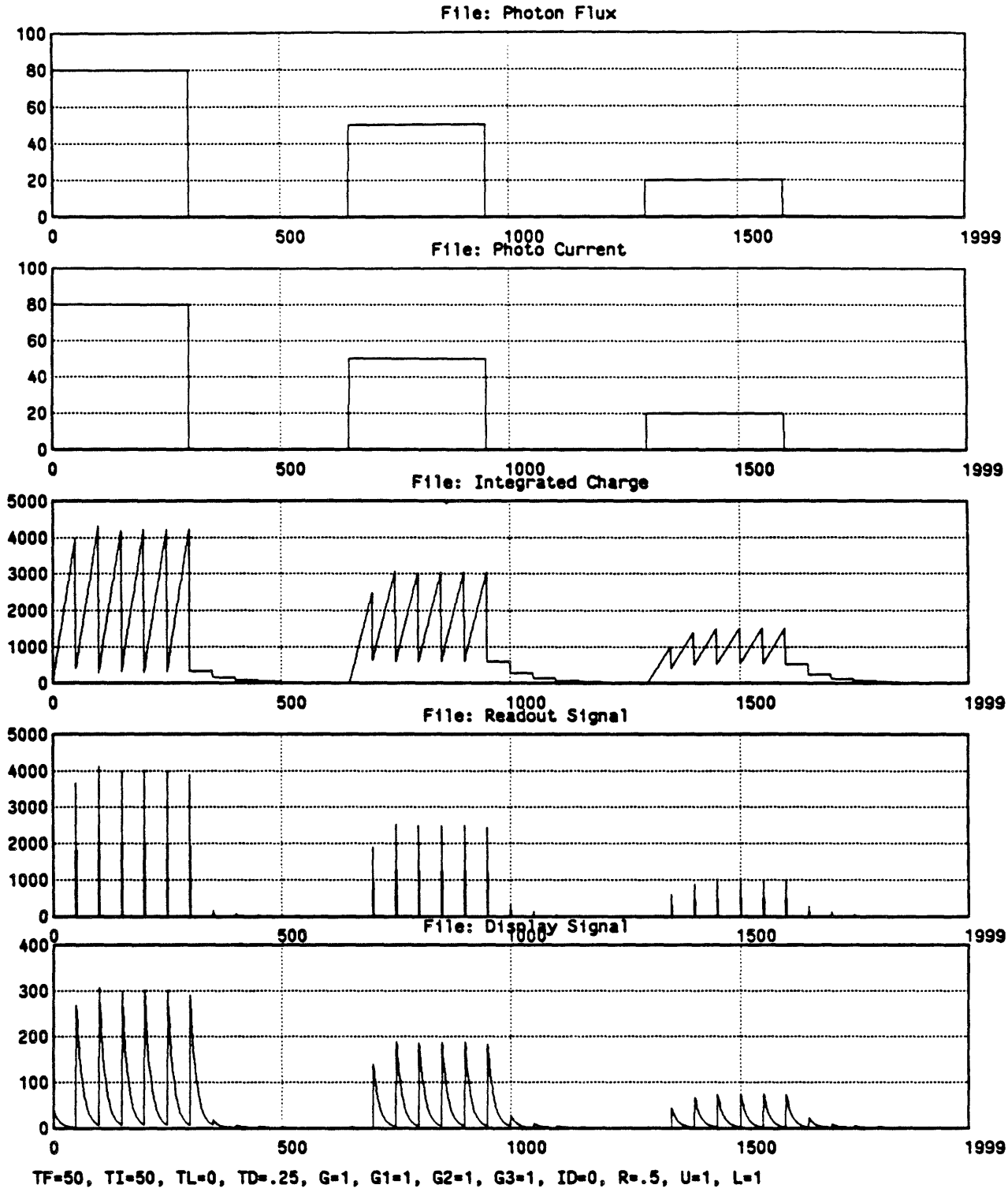


Fig. 3.5(j) Signal-dependent lag without bias light.

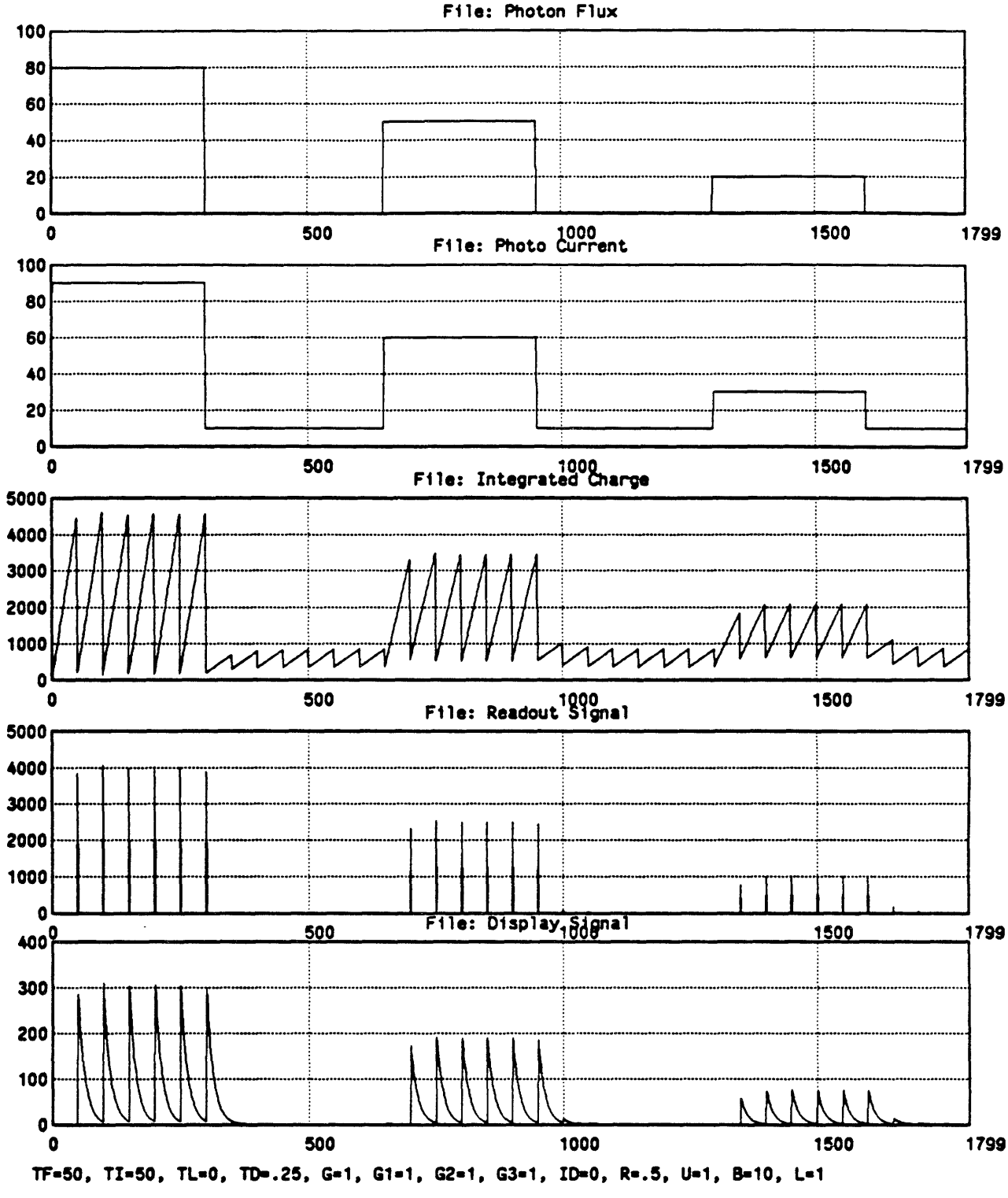


Fig. 3.5(k) Improvement of signal-dependent lag with 10 units of bias light.

Response to a periodic ramp of light. A ramp of light is useful for studying the tonal characteristics of imaging devices. If the camera has a gamma less than unity, the response to a linear ramp waveform will be nonlinear and will tend to compress the highlights. In Figs. 3.5(h) and (i), the light input is a ramp waveform with a period of 10 frames. A linear camera with beam discharge lag (90% readout) attached to a linear display element is shown in Fig. 3.5(h). The tonal characteristics are preserved in this situation, implying a unity gamma. In other experiments, the readout efficiency was varied between 50% and 100%, and the tonal characteristics were still preserved. Fig. 3.5(i) shows a lag-free camera with a gamma of $\gamma_I = 0.65$ connected to a display with a gamma of $\gamma_D = 2.2$. The entire system is not quite linear because it has an overall gamma of $0.65 \times 2.2 = 1.43$. This is typical of a vidicon camera attached to a CRT. This experiment illustrates that gamma originates in the light-to-charge mechanism, not in the charge readout process.

Signal-dependent beam lag and bias lighting. In previous experiments, the readout efficiency in the exponential model was independent of the available charge. When the readout efficiency decreases with available charge, signal-dependent lag can be modeled. This is illustrated in Fig. 3.5(j). This experiment models a linear camera with zero photoconductive lag. The charge readout mechanism has the following characteristics: $\eta_R = 0.9$ for $q_I \geq 5000$ charge units, and η_R linearly decreases towards 0.5 as q_I approaches zero. Signal-dependent readout efficiency produces a beam lag that is worse at lower levels of illumination. A common and effective method of improving lag performance is to add a small amount of bias light to the rear surface of the photoconductor. In Fig. 3.5(k), the photocurrent is artificially augmented by 10 units, and a proportional amount is subtracted from the output signal current. This bias of the charge pattern yields higher values of readout efficiency, and the beam readout lag is reduced, even at low signal levels.

Stability of linear and exponential readout models. As mentioned earlier, linear readout exhibits a linear lag response, and exponential readout exhibits an exponential lag response. Furthermore, the two models have different charge dynamics for a constant luminance input: the exponential response stabilizes while the linear response does not. Fig. 3.6 shows the exponential model stabilizing around a fixed

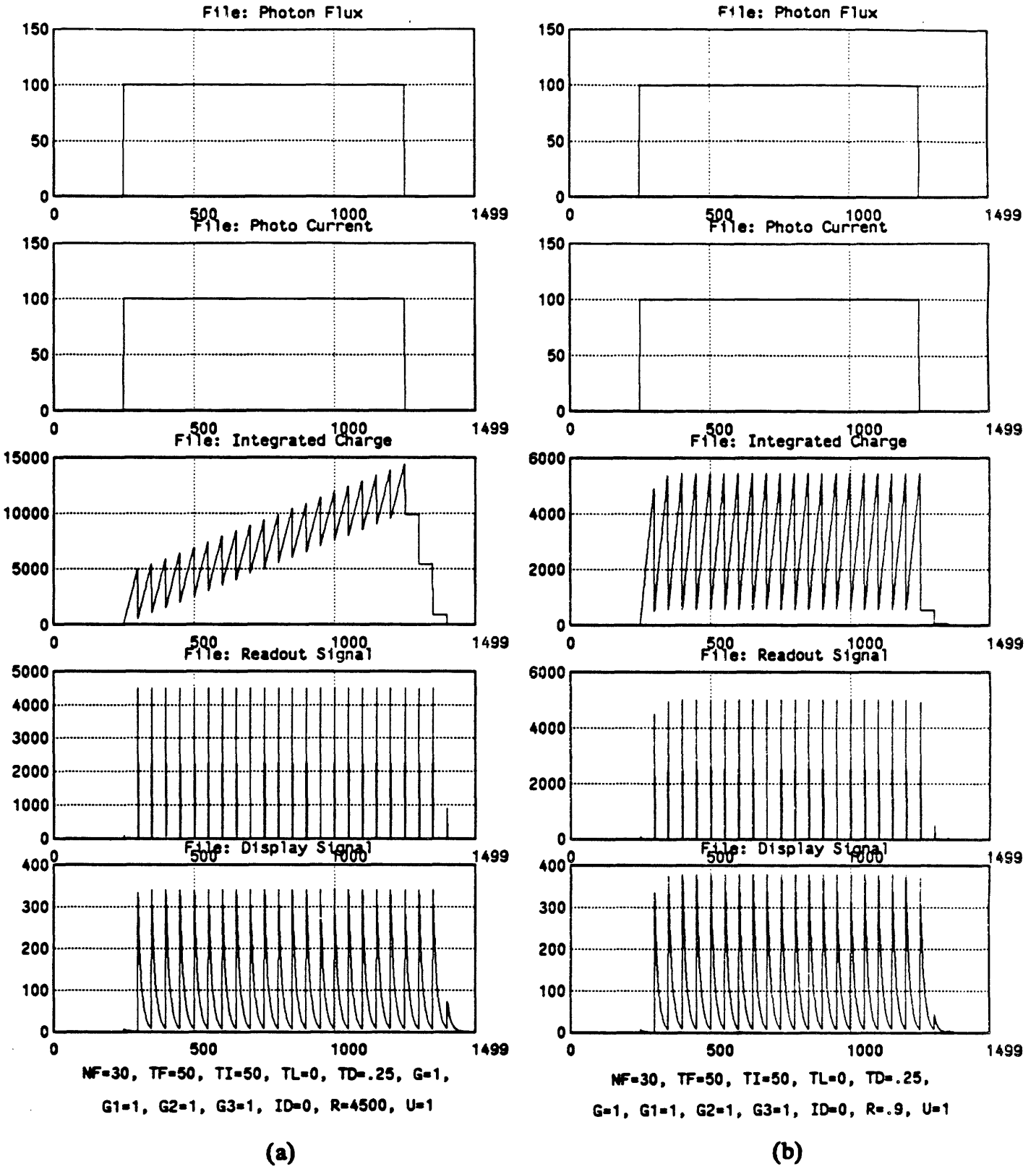


Fig. 3.6. Stability of charge dynamics under different readout mechanisms. (a) Linear readout of 4500 charge units. (b) Exponential readout with 90% efficiency.

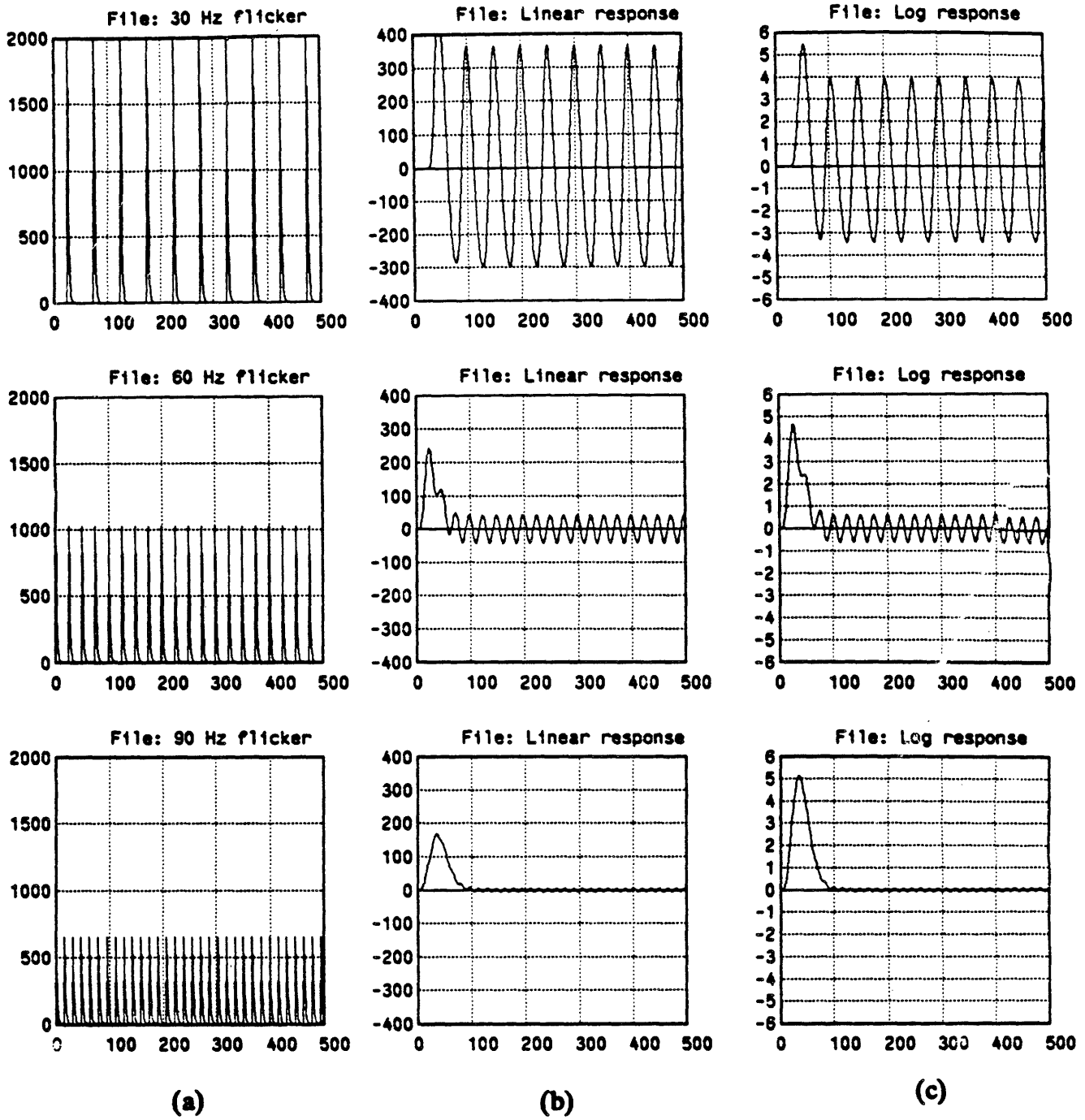


Fig. 3.7. Flicker perception at 30, 60, and 90 Hz. (a) Display output. (b) Linear HVS response. (c) Log HVS response.

operating point while the linear charge dynamics continue to rise. Because target stabilization is observed in practice, the exponential model is a better characterization of charge readout. The linear model can be made to stabilize, however, by introducing a threshold, or saturation, parameter.

Flicker perception. Fig. 3.7 shows the response to a small area on a CRT when the refresh rate is 30, 60, and 90 Hz, with and without logarithmic transformation in the HVS model. A discretized version of the temporal impulse response shown in Fig. 3.4(b) was used. The 60 Hz simulation can be interpreted as the output of a single element in a 60 Hz progressively scanned display, or as the output of two elements on physically adjacent lines in a 30 Hz interlaced system. Note the reduced response at 60 and 90 Hz, with and without logarithmic transformation before the linear HVS filter. This indicates that flicker would be barely perceptible at these rates, which is in agreement with observed results.

3.2. A 3-D Spatiotemporal Model of a Television System

The extension of the 1-D functional model to 3-D requires two main additions: (1) incorporating the correct inter-element timing relationships of all important photoelectric processes, and (2) modeling inter-element interactions, chiefly charge spreading in the camera. These features permit study of the motion rendition of many common imaging devices.

3.2.1. 3-D Temporal Integration Patterns

To model faithfully the motion rendition of imaging devices, it is necessary to characterize their temporal operation. Though all camera elements integrate light for the same amount of time, their integration periods may be *time-shifted.* This is illustrated for various imaging devices in Fig. 3.8. In each figure, temporal integration of an image of 12 lines and 16 elements per line is followed through time. Each camera element is shown as a small square in the xy-plane, and the "length" of the element in the temporal direction is proportional to the integration period. Furthermore, the integration boundaries are shown by the "front" and

"rear" surfaces of each element. Thus, it is easy to see exactly when each element starts and stops integrating the image illumination function, $e_C(x,y,t)$, in relation to its neighbors.

Each integration pattern is characterized by three parameters: T_F, the frame time, which is constant for all elements; T_I, the integration time, which is also constant for all elements; and T_O, a time offset, which may be different for each element in a given frame. The readout is assumed to be instantaneous ($T_R = 0$) and simultaneous with the start of the next frame period. In most imaging devices, T_I is equal to the frame time, T_F, or the field time, $T_F/2$. However, some modern shuttered or solid-state imaging devices may have a variable integration time [84]. To show this generality, all integration patterns have thick black bars for the elements in the upper and lower right corners of the image. This bar illustrates a variable integration interval for the imaging device.

Fig. 3.8(a) shows the temporal integration pattern for movie film or a progressive frame-transfer CCD. All elements integrate light *simultaneously;* therefore, they have the same time offset. In a movie camera, the angular width of the shutter blade determines T_I. A CCD operating in this non-interlaced mode may be called a progressive frame-transfer device. Some CCD's operate in this mode and simulate interlace by reading out the integrated lines in line-interlaced order.

Fig. 3.8(b) shows the temporal integration pattern of a progressive-scan tube-type camera. Scanning begins in the upper left corner of the image. The effective erasure width of electron beam is assumed to be equal to the line width. The time offset for each element along a scan line is a linear function of the x-position of the read beam. This explains why the element boundaries along a scan line are skewed forward in time from left to right. For unshuttered progressive-scan camera tubes, $T_I \approx T_F$. Beam retrace time in both directions is assumed to be zero.

Fig. 3.8(c) shows the temporal integration pattern of a progressive-scan line-transfer CCD. Since there is no shutter, all elements integrate charge continuously except during the short amount of time it takes to transfer a line of charge to a storage register. Thus, $T_I \approx T_F$. Lines are read out, one at a time, from top to bottom. This pattern assumes zero blanking time between frames. The time offset is

constant for all elements along a scan line, but linearly increases from one line to the next. Except for special cases, one would expect similar motion rendition from temporal integration patterns (b) and (c).

Fig. 3.8(d) shows the temporal integration pattern of an interlaced line-transfer CCD [85, 49]. The pattern for an interlaced camera tube would look very similar, if the beam width were small enough. However, as is the case with most interlaced camera tubes [86], the entire target is erased every field, thus $T_I = T_F/2$.

Fig. 3.8(e) shows the temporal integration pattern of an interlaced interline-transfer CCD [85, 87]. All the elements in the odd field have a constant time offset, $T_O = 0$; all those in the even field have a different offset, $T_O = T_F/2$. Each element may integrate up to the full frame period.

Fig. 3.8(f) shows the temporal integration pattern of an interlaced frame-transfer CCD [88]. The effective height of each element is twice its width, but the element centers shift from field to field, simulating interlace. This line averaging degrades the vertical resolution in a given field, but this effect is somewhat compensated by the phasing between fields. All elements have the same time offset: $T_O = 0$ for the first field, and $T_O = T_F/2$ for the second. Each element may integrate up to half the frame period.

These figures illustrate that the temporal integration functions of film vs. video and of interlaced vs. progressive scan are quite different. Since these functions affect motion rendition, it is instructive to model them and study their effects.

The temporal interpolation patterns of common display devices are very similar in shape to some of the temporal integration patterns of Fig. 3.8. Each integration period can be thought of as an interval of luminous flux emanating from a specific display element. Neglected here is the relative intensity of the luminous flux as a function of time. In CRT's, for instance, the pattern of light emanating from each display element can be modeled by a rapidly decaying exponential.

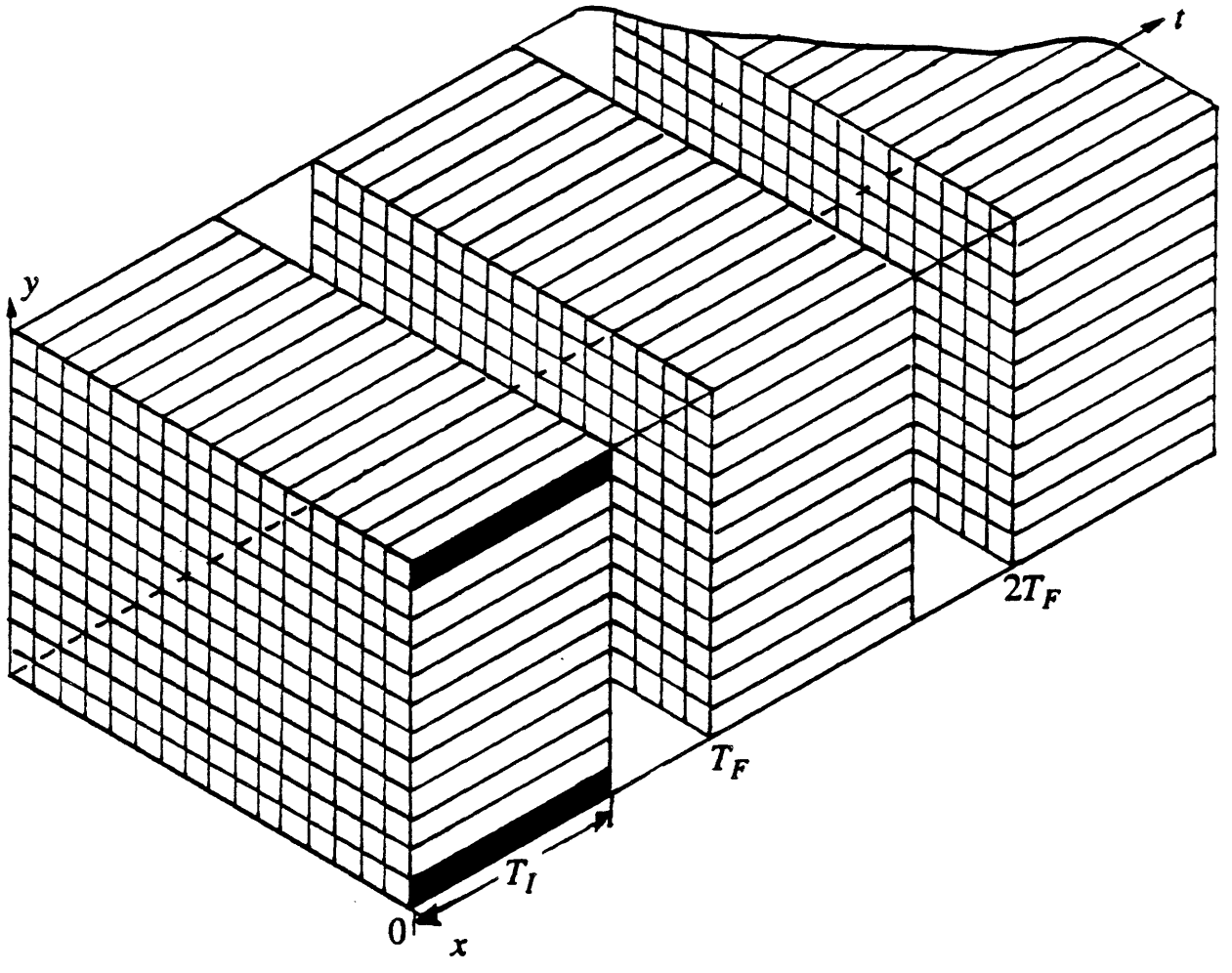


Fig. 3.8(a). Movie film or progressive frame-transfer CCD.

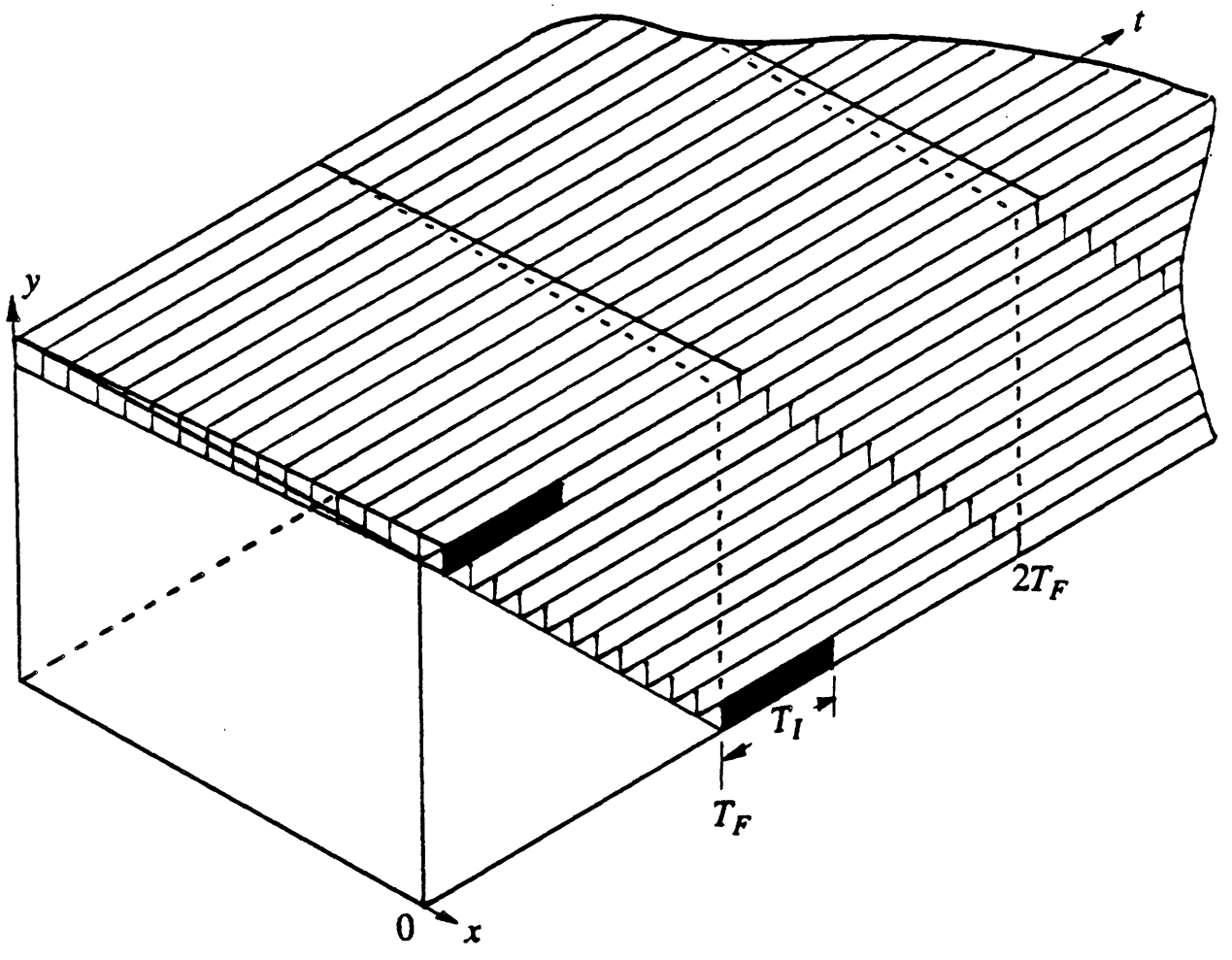


Fig. 3.8(b). Progressive tube-type video camera.

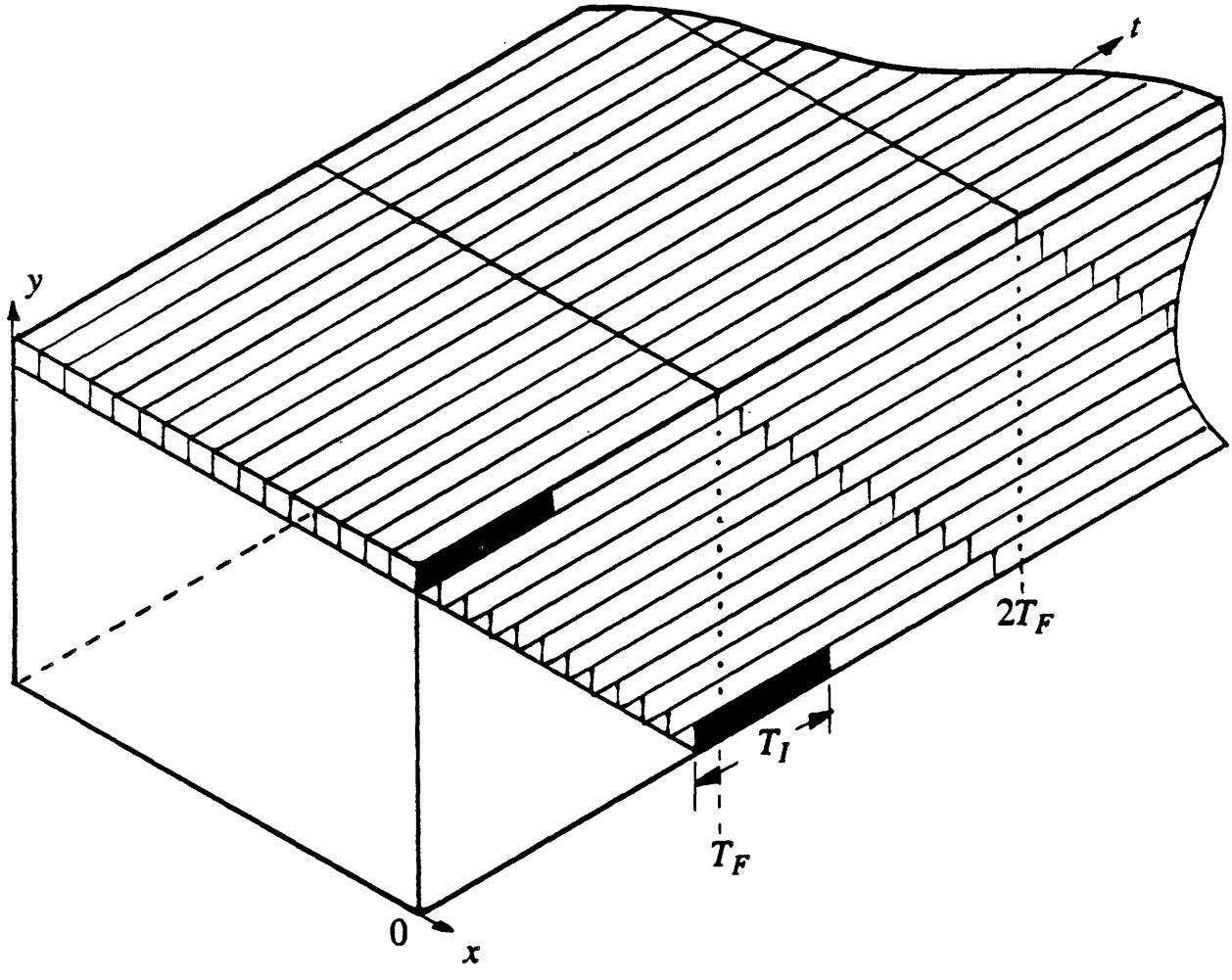


Fig. 3.8(c). Progressive line-transfer CCD.

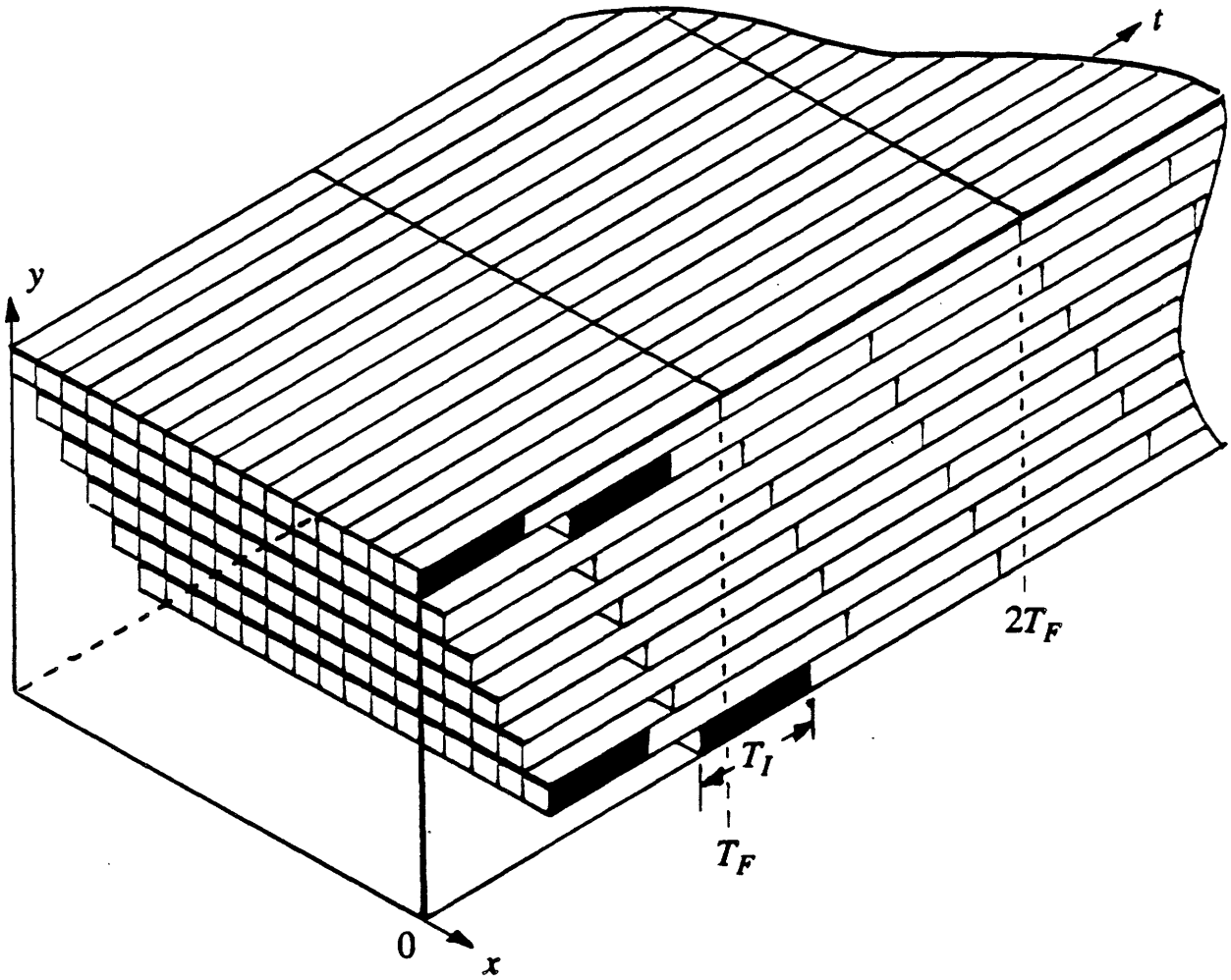


Fig. 3.8(d). Interlaced line-transfer CCD.

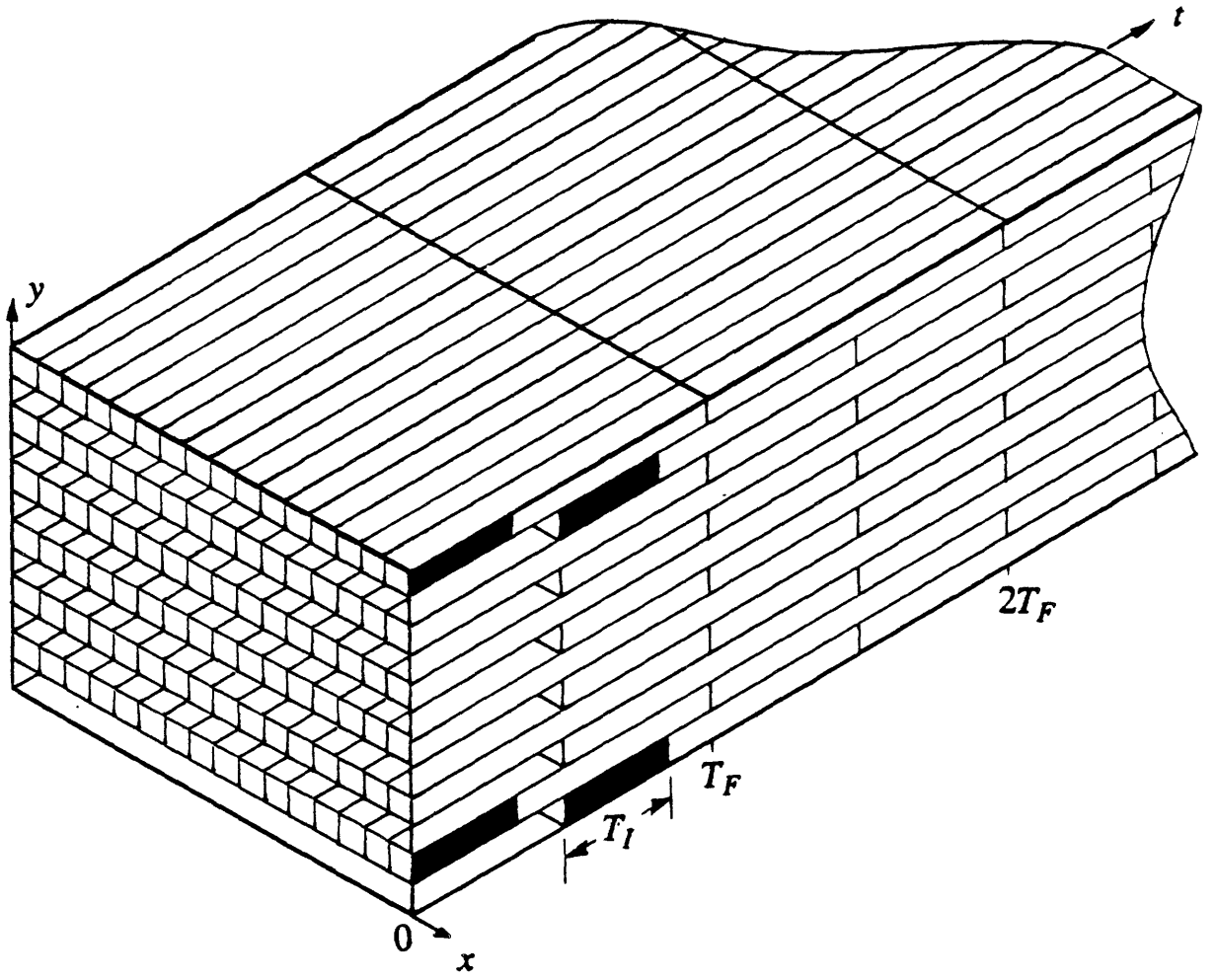


Fig. 3.8(e). Interlaced interline-transfer CCD.

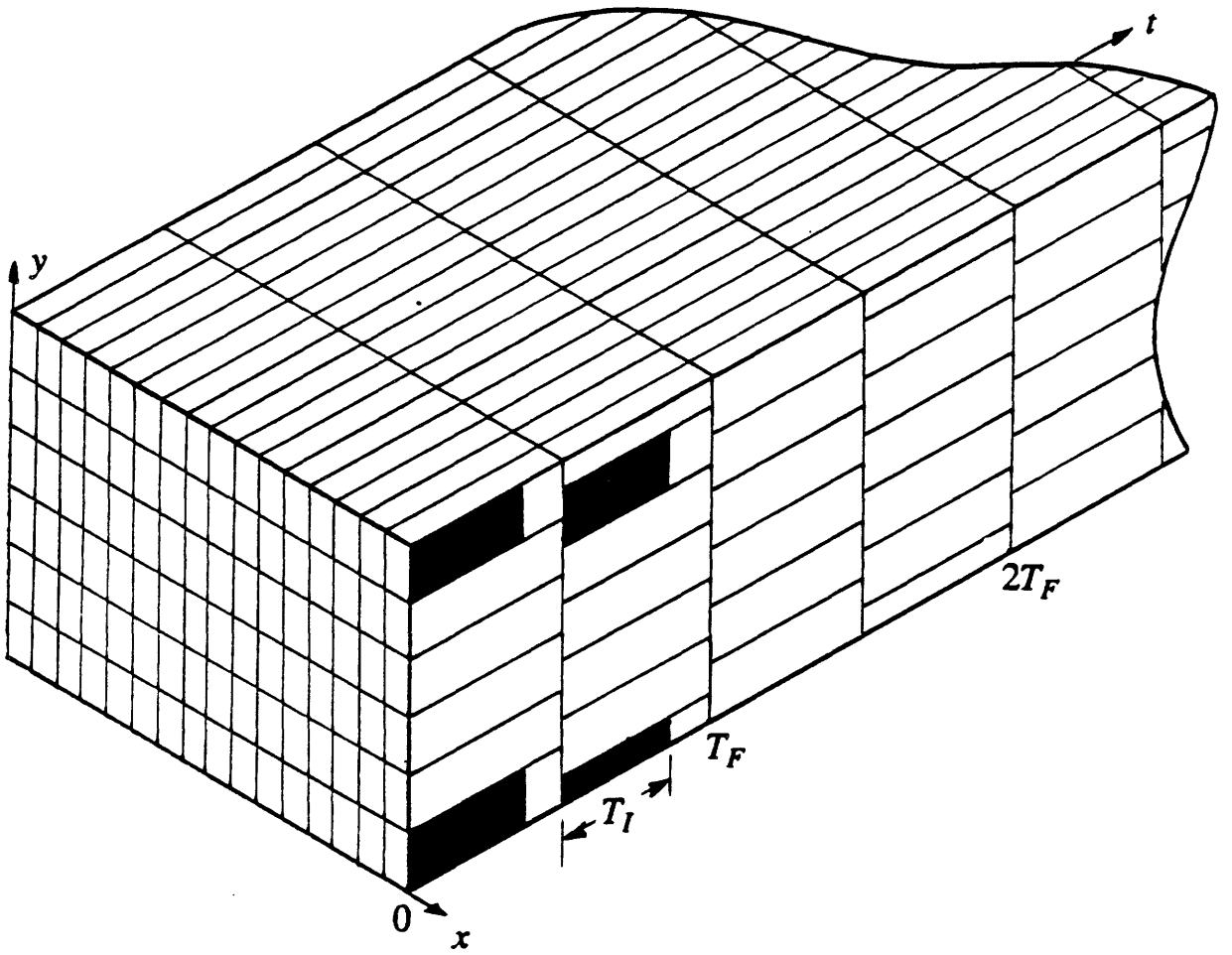


Fig. 3.8(f). Interlaced frame-transfer CCD.

3.2.2. Modeling Charge Spreading

The 3-D functional model developed so far ignores the effects of neighboring elements. In reality, some imaging devices, such as vidicons, must be carefully designed so that inherent charge spreading is not a major degradation to spatial resolution [89].

A model characterized by a single time constant is sufficient to analyze charge spreading effects. Assume that the photosensitive surface can be characterized by an effective resistance per unit area, R, which is independent of the lateral flow of current. Furthermore, let C be the capacitance per unit area between the front and rear surfaces of the photosensitive film. Under these assumptions, $V(x,y,t)$, the voltage distribution on the photoconductor with respect to the cathode satisfies the *heat equation* [90, 91]

$$\nabla^2 V = RC\frac{\partial V}{\partial t}. \tag{3.17}$$

A discrete approximation to this equation is [92]

$$V(i,j,k) = \frac{RC}{4+RC}\cdot V(i,j,k-1) \tag{3.18}$$

$$+ \frac{1}{4+RC}\cdot[V(i+1,j,k)+V(i,j+1,k)+V(i-1,j,k)+V(i,j-1,k)],$$

where i,j, and k are the discrete horizontal, vertical, and temporal variables. Note that the value at time k is the sum of two contributions: the value at time $k-1$, and an average of the neighboring values at time k. The relaxation time constant, RC, determines the importance of each term. To minimize blurring due to charge relaxation, a relatively long time constant is desired. This implies that $V(i,j,k)\approx V(i,j,k-1)$. If instead $RC\approx 0$, local averaging would dominate the process, causing a rapid loss of spatial resolution.

3.2.3. Experiments

Processing Discretized Image Sequences. The programs *int* and *rec* simulate the spatiotemporal characteristics of a camera mosaic connected by a noiseless channel to a display mosaic. The program *int* models the integration and scanning functions in the camera, and *rec* models the scanning and reconstruction functions in the display. The size of the camera aperture is assumed to cover exactly one element, so the spatial filtering characteristics of the aperture are not explicitly considered. Time is expressed in frame periods. The smallest time increment is T_H, the time required to scan a single horizontal line of charge. Each simulation spans a number of *full frames.* Each full frame spans an integral number of *subframes;* furthermore, each subframe spans an integral number of *line times.* The slowest simulation rate corresponds to one scan line per subframe; the fastest simulation rate corresponds to N_V scan lines per subframe, where N_V is the vertical size of the camera mosaic.

The program *int* processes discretized time samples of the 3-D image illuminance function, $e_C(x,y,t)$. It produces two outputs: $q_I(x,y,t)$, the charge pattern on the scanned surface of the photosensor, and $v(t)$, the gamma-corrected video signal. The program *rec* operates on $v(t)$ to simulate the 3-D luminance function of the display, $l_D(x,y,t)$. This waveform can be interpreted as snapshots of the screen of a hypothetical display. Each snapshot lasts for N_L line times. The spatiotemporal processing of $l_D(x,y,t)$ by the HVS determines the perceived resolution and motion rendition of the result.

The following experiment illustrates important system waveforms associated with various camera and display characteristics. In Fig. 3.9, the image illuminance function is shown by the uppermost row of subframes. Successive rows show the luminance function of the display; each is defined by a different set of parameters.

The input flux, Fig. 3.9(a), consists of 16 subframes, each of size 16×16. Strobed images in the shape of a square and a cross appear in subframes 1 and 9. In each case, the number of scan lines per frame and the interlace ratio are the same for the camera and display. Figs. 3.9(b) and (c) show a system with a scan rate of 1 scan per subframe, progressive. Thus, the entire row spans one full

frame. The camera is linear, has zero lag, and has 100% readout efficiency. The display is linear with an exponential phosphor decay. The time constant is 0.25 frames in (b) and 0.5 frames in (c). Note that the square and the bottom half of the cross appear together on the display; this is because the scan rate is too slow. Figs. 3.9(d) and (e) are the same as (b) and (c) except that the scan rate is twice as high. The charge image of the square is completely erased before the image of the cross is generated. The partial overlap of the square and the cross in frames (10) through (14) in Fig. 3.9(e) is due to the phosphor persistence of the display, and is not a camera artifact. The scan rate doubles in (f), (g), and (h), yielding a rate of 4, 8, and 16 scan lines per subframe. The phosphor decay rate is 0.5 full frame periods in each case. Although (h) is physically quite similar to the input sequence, it must be remembered that *all* the display sequences (b) through (h) may *look* the same to a human observer, depending on the actual scanning rate and viewing conditions. In Figs. 3.9(i), (j), and (k), camera lag has been introduced. Photoconductive lag with a time constant of 0.75 frame periods produces the lag in (i). A readout efficiency of 75% produces the beam discharge lag in (j), and the presence of both lag processes produces the effect in (k). In Figs. 3.9(l), (m), and (n), both lag processes have been modeled, in addition to charge spreading effects. The charge spreading time constant is 10, 5, and 1 frame period, respectively. Note the significant blurring that occurs in (n). In Figs. 3.9(o) and (p) the effects of interlace are modeled. All other parameters are the same as in (k). In (o) there are 8 scan lines per subframe with 2:1 interlace; in (p), there are 4 lines per subframe with 4:1 interlace. These results may be compared to those of later experiments that simulate object motion.

Processing Analytic Image Illuminance Functions. The program *tip* simulates the effects of motion rendition for various imaging devices. Rather than processing a given sequence of image frames, *tip* uses an *analytical model* of the image illuminance function. That is, $e_C(x,y,t)$ is mathematically defined for any floating-point triplet (x,y,t). Analytic 3-D functions can be defined to represent the motion of simple objects, such as rectangles or ellipses. These functions can then be integrated both spatially and temporally with very fine step sizes to simulate any of

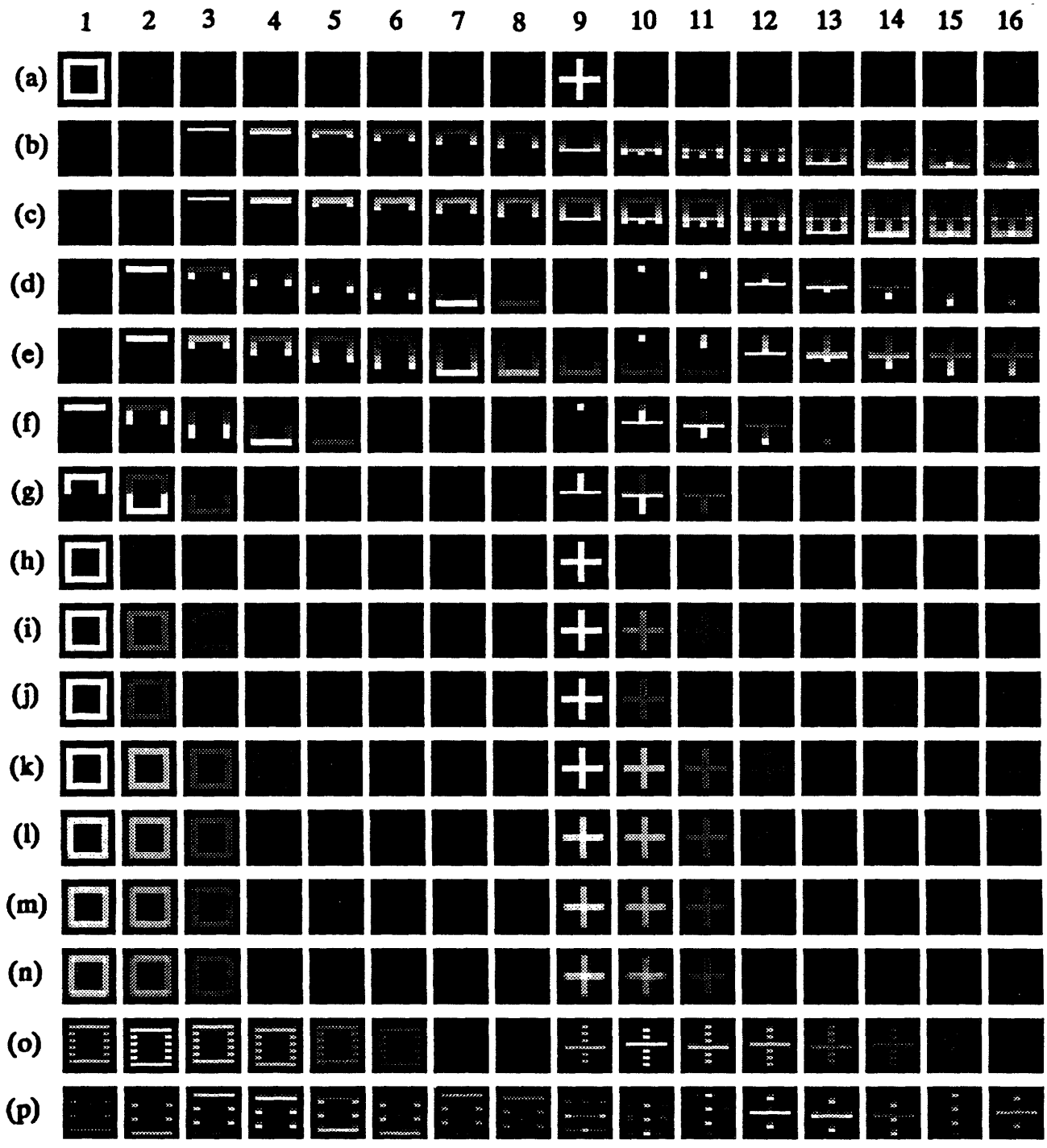


Fig. 3.9. Display snapshots of a 3-D TV simulation.

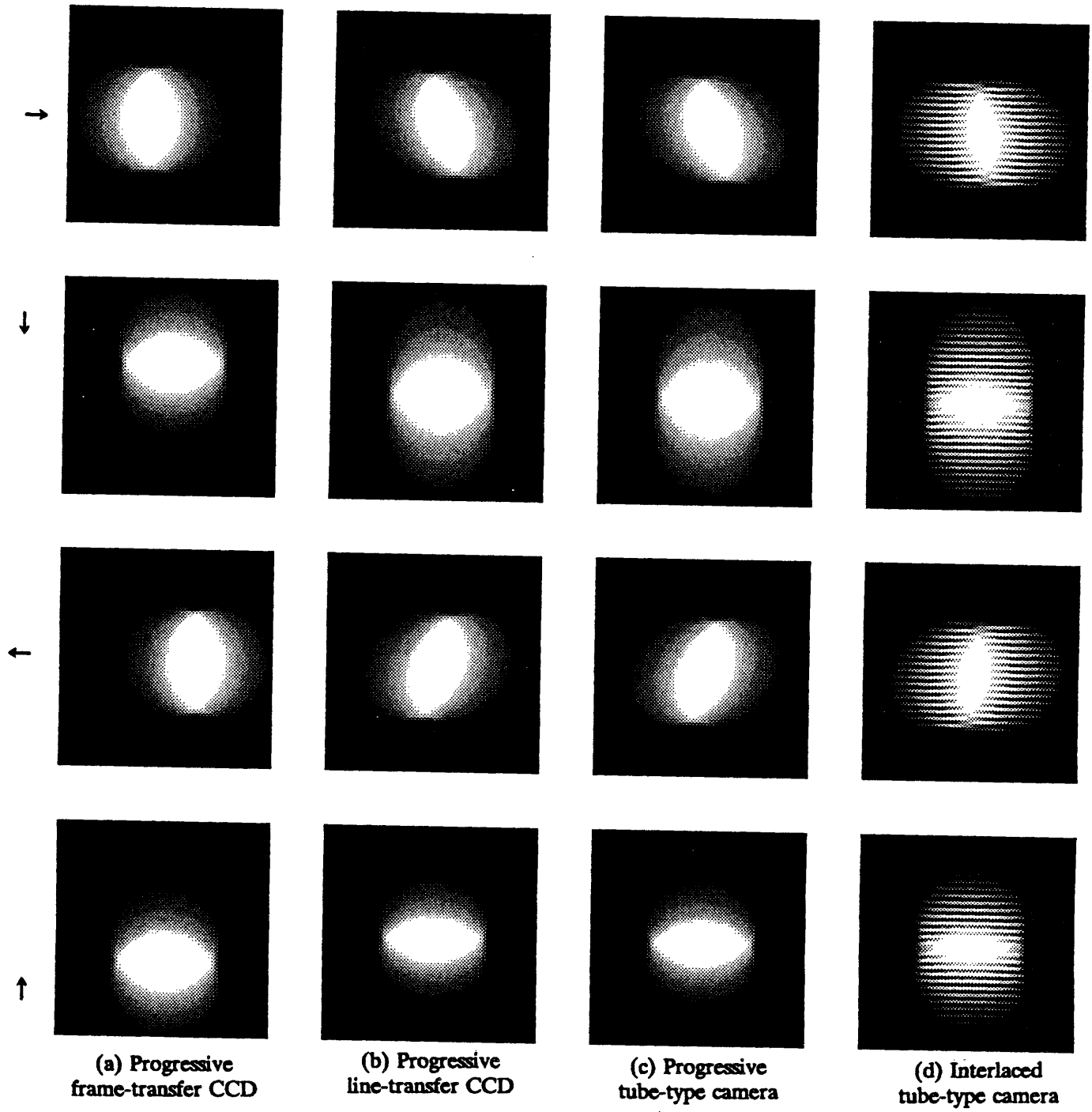
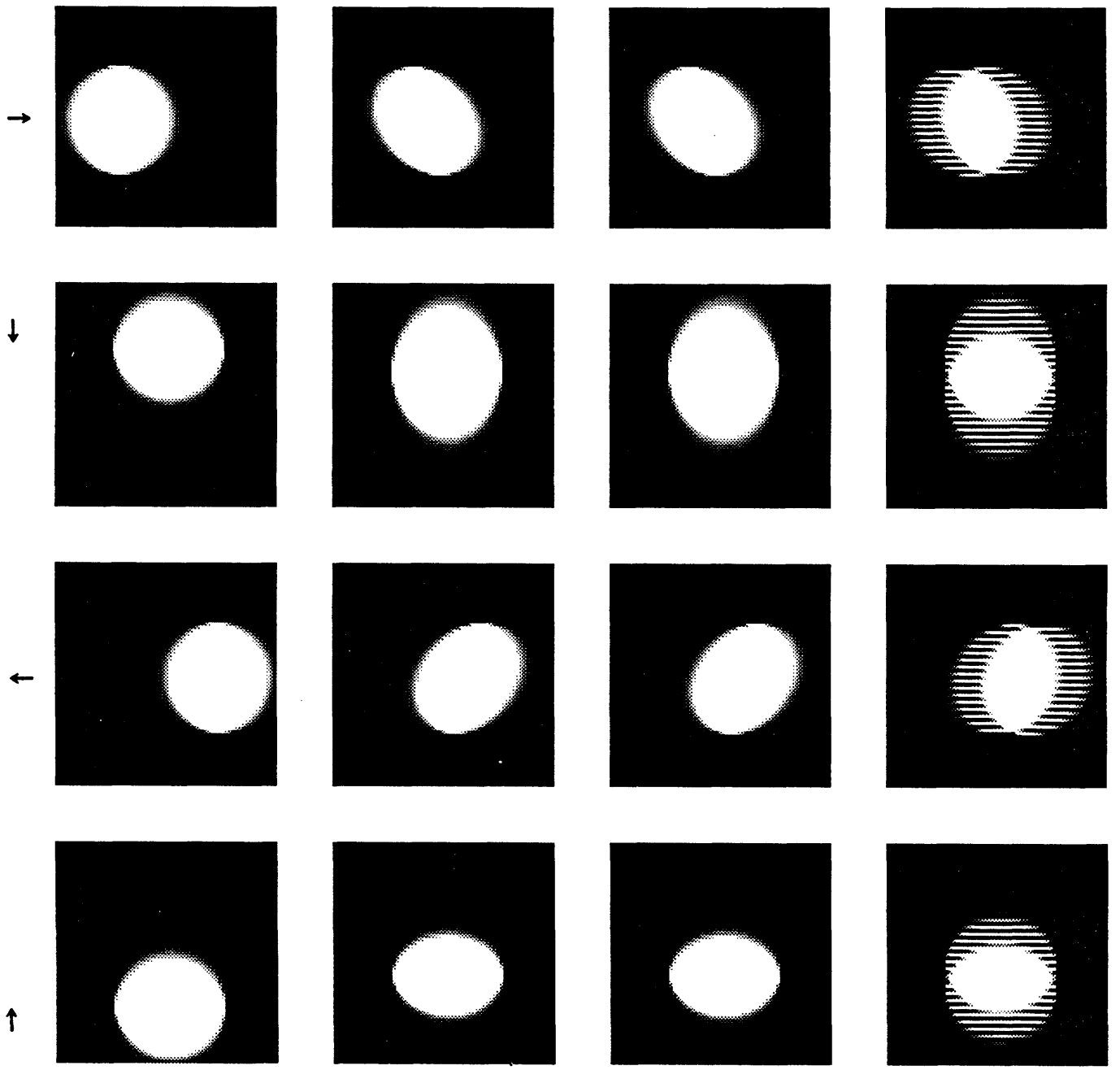


Fig. 3.10. Motion rendition of moving disk. $T_I = T_F$.



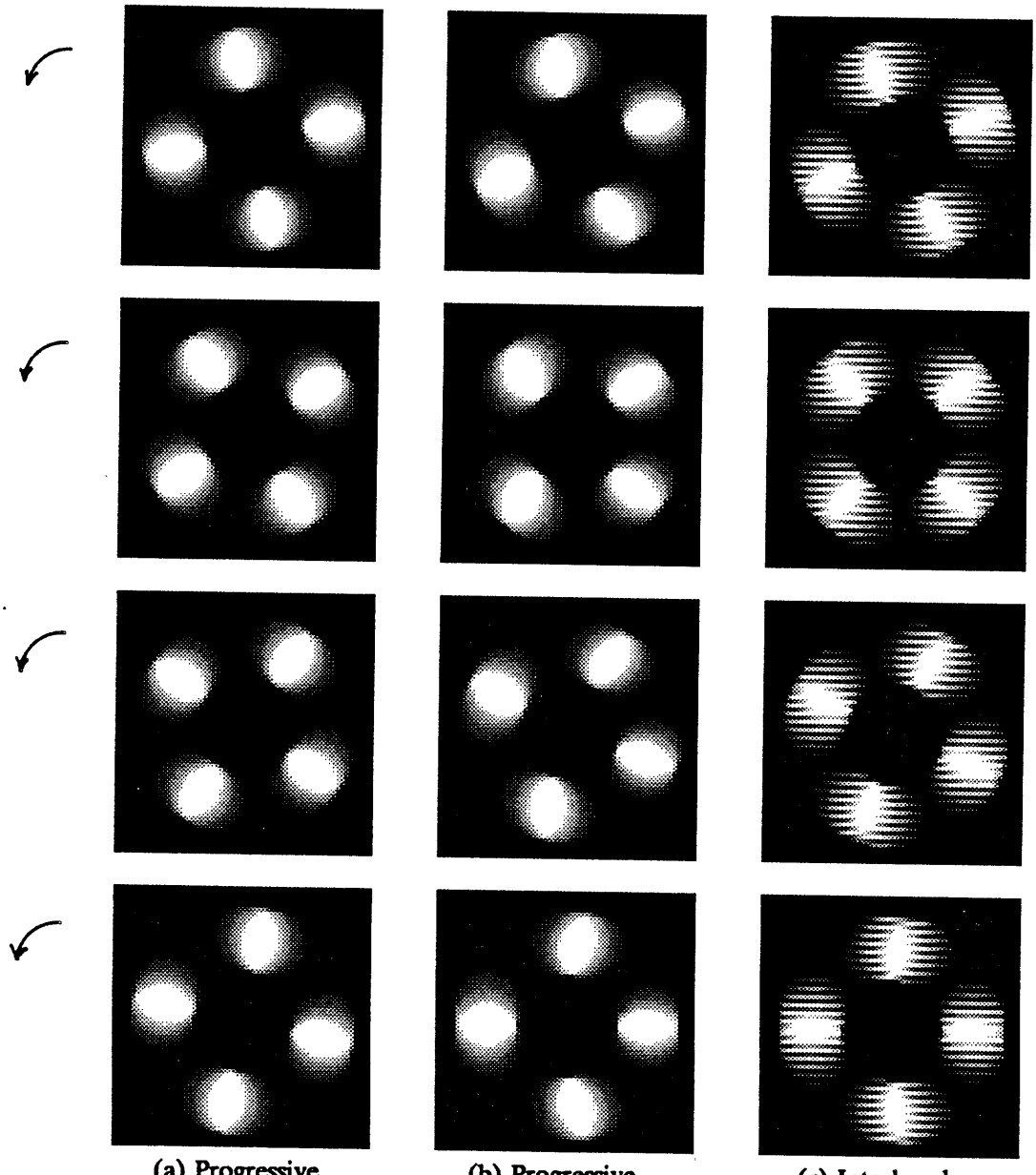
(a) Progressive frame-transfer CCD

(b) Progressive line-transfer CCD

(c) Progressive tube-type camera

(d) Interlaced tube-type camera

Fig. 3.11. Motion rendition of moving disk. $T_I = 0.25 \cdot T_F$.



(a) Progressive frame-transfer CCD

(b) Progressive line-transfer CCD

(c) Interlaced tube-type camera

Fig. 3.12. Motion rendition of revolving disks. $T_I = T_F$.

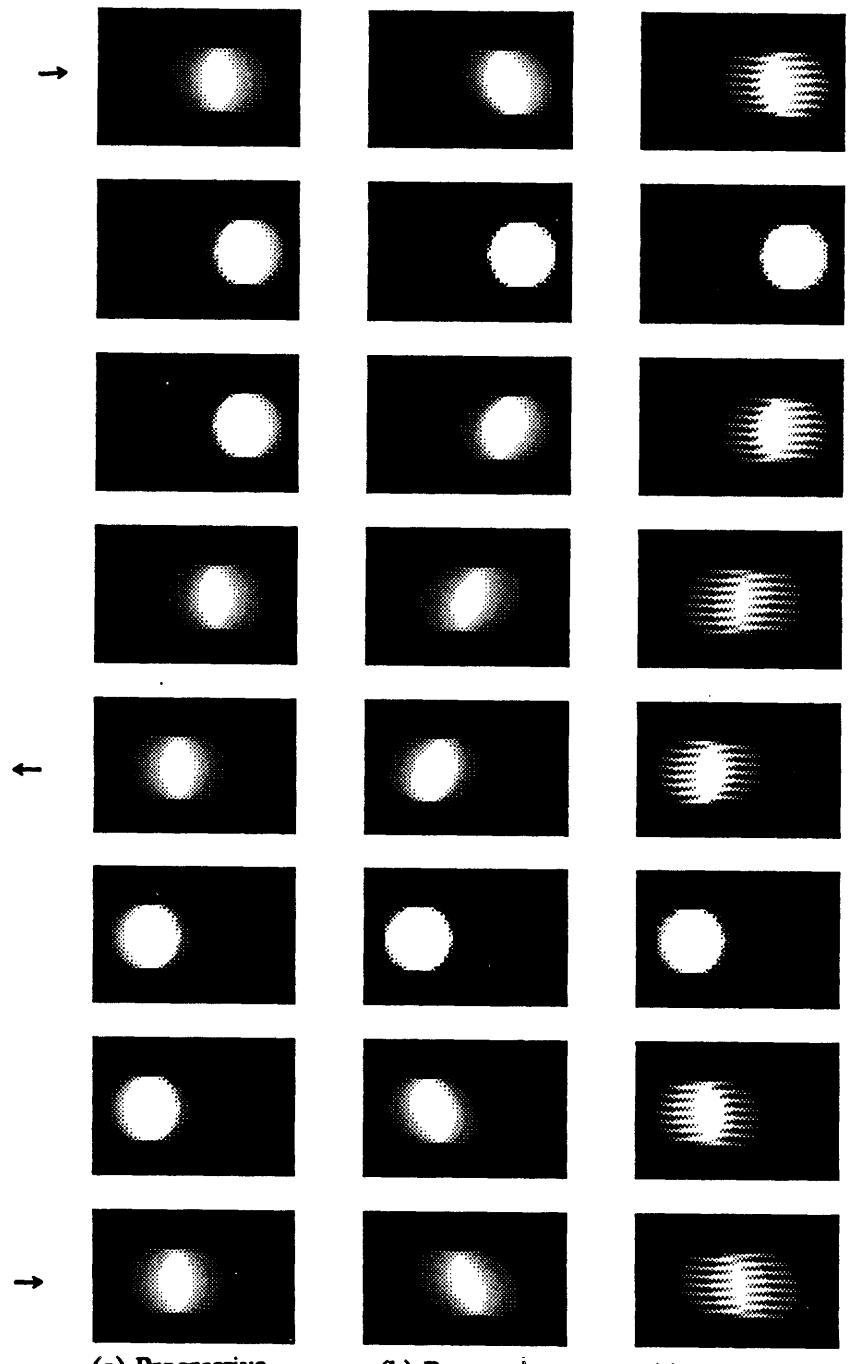


Fig. 3.13. Motion rendition of oscillating disk. $T_I = T_F$.

the integration patterns shown in Fig. 3.8. The results of several motion rendition experiments are shown in Figs. 3.10-3.13. These images should be thought of as photographs taken of the screen of an ideal display. Fig. 3.10 shows a uniformly illuminated disk moving at constant velocity across the field of view. The velocity is 0.25 picture widths per frame period. Motion is left-to-right along the top row, top-to-bottom along the second row, right-to-left along the third row, and bottom-to-top along the bottom row. For this simulation, $T_I = T_F$. Note the skewing and stretching of the disk that occurs in the last three columns. This type of deformation is more apparent in Fig. 3.11, where the integration period has been shortened: $T_I = 0.25 T_F$. Note that the shape of the deformation depends on the speed and direction of the moving object. Motion tracking algorithms must take into account this deformation. Fig. 3.12 shows four successive frames of four revolving disks. The angular velocity of each disk is 22.5° per frame period. Note the different deformation experienced by each disk within any frame in the second and third columns. In this simulation, $T_I = T_F$. Fig. 3.13 shows a disk oscillating horizontally with a sinusoidal motion. The period of oscillation is $8T_F$. Note that the severity of deformation (or field separation) depends on the speed of the object. For this simulation, $T_I = T_F$.

3.3. Summary

This chapter examined the temporal processes of a single image element. It was shown how camera lag mechanisms arising from different physical processes can produce similar video signal waveforms. Two models of charge readout were proposed and simulated. The exponential model accurately simulates the exponential lag behavior of many tube-type cameras and it also yields stable charge dynamics; it can also be extended to model signal-dependent lag. The 1-D model included a temporal model of the HVS, which predicted decreased flicker sensitivity at 60 and 90 Hz. A simple 3-D television model was developed by considering the temporal integration patterns of all elements as well as charge spreading between elements. The motion rendition of various imaging devices was simulated. It was shown that in some cases the shape of a moving object deforms, and the severity of deformation depends on the object's velocity.

Chapter 4

Adaptive Image Interpolation for Television

4.0. Introduction

The previous chapters developed a numerical model for the conventional television process. This chapter extends the model by incorporating "smart" analog and digital processing for enhancement of displayed images. Two adaptive methods of image interpolation are described. Both are designed to reduce line structure visibility and aliasing, but are best suited for different applications. The first is an analog beam shaping technique that requires no increase in monitor bandwidth; the second is a digital interpolation algorithm that can be used in a motion-adaptive interlaced-to-progressive scan converter.

Sampling and interpolation are fundamental to television. A video camera samples a 3-D image illuminance function, $e_C(x,y,t)$, to form a 1-D signal; a video monitor interpolates the 1-D signal back into a 3-D luminance function, $l_D(x,y,t)$. Aliasing, the result of improper sampling, must be minimized prior to transmission. Once it occurs, aliasing can be reduced, but usually at a loss of resolution. Sampling structure visibility, the result of imperfect interpolation, can theoretically be eliminated by ideal low-pass filtering. As shown in Chapter 2, at a viewing distance where line structure is not perceived, aliasing may still be quite visible and therefore can be the most significant source of image degradation [93]. In television displays, physical constraints limit the quality of image reconstruction and cause the ubiquitous tradeoff between line visibility and spatial blur.

Experiments have shown that decreasing the line visibility improves the subjective appearance of displayed images [94]. The simplest and most obvious method of decreasing line visibility and aliasing is to increase the size of the spot; however,

this blurs the reconstructed image [69]. In a conventional monochrome CRT, a circular Gaussian spot sweeps across the phosphor-coated surface of the display in a raster pattern consisting of closely spaced horizontal lines. Essentially, the beam filters horizontally and interpolates vertically. The spot must be large enough to minimize line visibility, yet small enough to retain picture detail.

A possible improvement to the conventional method of scanning is to change the Gaussian spot shape adaptively at each point on the scan line. This is technically challenging because it requires dynamic beam shaping at megahertz rates; however, dynamic focusing at lower rates is currently in use [68, 95]. Assuming the feasibility of dynamic Gaussian beam shaping, several questions can be posed: can the quality of an image be improved by dynamically adjusting the beam shape at each point on the raster? What information determines the beam shape at each point? Can the computation of beam shape parameters be done in real time? The first section in this chapter offers some insight into these problems.

Another method of reducing line visibility is to digitally interpolate new lines between available ones and to use a smaller Gaussian spot. This method requires a high-bandwidth video monitor with stable high-frequency scanning circuits. Conventional interpolation methods are intrafield line averaging and motion-adaptive interpolation [96]. A drawback of these methods is the aliasing produced along moving inclined edges and other contours. If adaptive contour interpolation were used in the interpolation algorithm, better results could be expected. The second section in this chapter develops a block-matching scheme for contour interpolation that greatly reduces aliasing artifacts.

4.1. Analog Gaussian Beam Shaping

As shown in Chapter 2, the interpolation of gamma-corrected video signals on a display screen can be modeled by a linear filtering operation. The spatial impulse function of a monochrome CRT is approximately Gaussian, and dynamic focusing circuitry can be used to insure that the beam shape remains circular over the entire scanning raster. In this section, the beam is modeled as a 2-D Gaussian with an

elliptical cross section. Because the beam shape is allowed to vary from point to point, linear shift-variant filtering must be used to describe the interpolation process.

In the following sections, linear shift-variant filtering for discrete 2-D signals is described in a general sense so that it may be used to interpolate in a single dimension or in both dimensions. Then, the computation of adaptive filter parameters (orientation and eccentricity of a 2-D Gaussian spot) is discussed. Attention is given to simple edge-finding algorithms that can be computed in real time with a minimum of video storage. The performance of these algorithms is investigated, and some simulation results are presented.

4.1.1. Linear Shift-Variant 2-D Filtering

Consider the discrete spatial model of the television process shown in Fig. 4.1. The input image $u(n_1,n_2)$ is filtered and subsampled by the video camera. As discussed in Chapter 2, subsampling occurs in both directions in solid-state arrays, but occurs only vertically in tube-type cameras. The video samples are ordered and sent over a channel. In a CRT, the video samples are filtered and interpolated vertically by the write beam. Conceptually, this is equivalent to filtering a zero-padded array of video samples. The following analysis concentrates on the filtering of the zero-padded array $x(n_1,n_2)$ to produce the interpolated array $y(n_1,n_2)$.

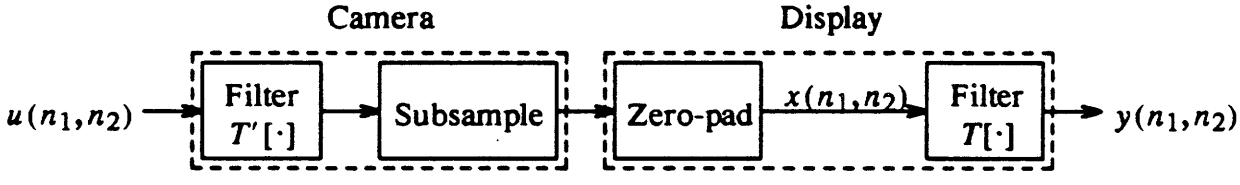


Fig. 4.1. Discrete model of the television process.

The most general description of a 2-D filter is given by [81]

$$y(n_1,n_2) = T[x(n_1,n_2)] = \sum_{k_1}\sum_{k_2} h(n_1,n_2;k_1,k_2;x(k_1,k_2)), \tag{4.1}$$

where $T[\cdot]$ is a transformation operator and h is the unit sample response at location (n_1,n_2) due to a unit sample at location (k_1,k_2) with possible amplitude dependence on the input $x(n_1,n_2)$.

If the system is *linear,* then the impulse response is independent of the input amplitude, and (4.1) becomes

$$y(n_1,n_2) = L[x(n_1,n_2)] = \sum_{k_1}\sum_{k_2} x(k_1,k_2)\cdot h_{k_1k_2}(n_1,n_2). \tag{4.2}$$

This is the form of a *linear, shift-variant* filter, where $h_{k_1k_2}(n_1,n_2)$ is the response of the linear system to a 2-D unit impulse located at (k_1,k_2).

To preserve the appearance of lines and edges, images are usually filtered with zero-phase finite impulse response (FIR) filters [97]. Assume that each $h_{k_1k_2}(n_1,n_2)$ has a finite region of support (ROS) of size $(2M_1+1)\times(2M_2+1)$, centered at (k_1,k_2). Several of these FIR filters are shown in Fig. 4.2. Notice that the ROS of each impulse response has the same *size* and *shape,* but is centered at a different *location.* Conceptually, one can think of the entire set of impulse responses as shifted versions of filters centered at the origin. The (k_1,k_2) location of the filter can be thought of as an *index:*

$$h_{k_1k_2}(n_1,n_2) = g_{k_1k_2}(n_1-k_1,n_2-k_2), \tag{4.3}$$

where $g_{k_1k_2}(\cdot,\cdot)$ is a set of impulse responses centered at the origin. Substituting (4.3) into (4.2) and inserting the correct summation limits yields

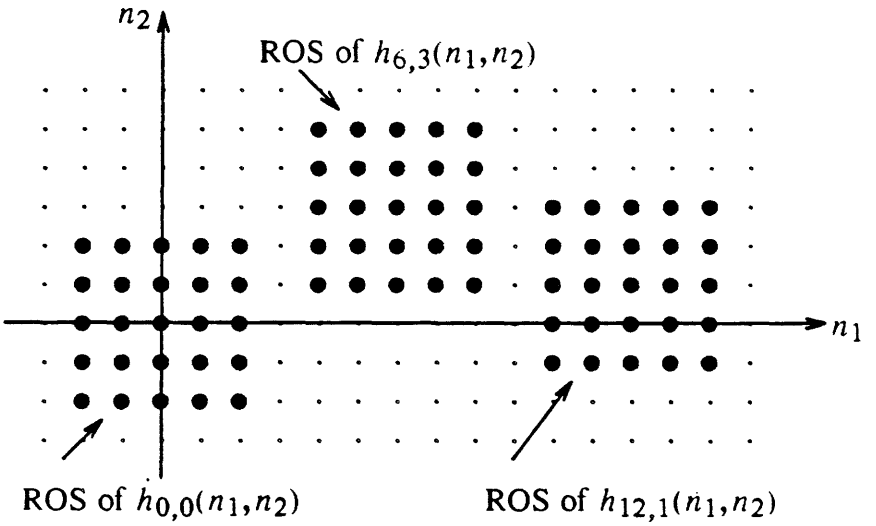


Fig. 4.2. Linear shift-variant FIR filters (ROS size: 5×5).

$$y(n_1,n_2) = \sum_{k_1=n_1-M_1}^{n_1+M_1} \sum_{k_2=n_2-M_2}^{n_2+M_2} x(k_1,k_2) g_{k_1,k_2}(n_1-k_1,\, n_2-k_2). \tag{4.4}$$

With $l_1 = n_1 - k_1$ and $l_2 = n_2 - k_2$, (4.4) becomes

$$y(n_1,n_2) = \sum_{l_1=-M_1}^{M_1} \sum_{l_2=-M_2}^{M_2} x(n_1-l_1,\, n_2-l_2) g_{n_1-l_1,n_2-l_2}(l_1,l_2). \tag{4.5}$$

Eq. (4.5) describes linear shift-variant filtering in its most general form. Each output point is obtained as the weighted sum of coefficients from $(2M_1+1)\times(2M_2+1)$ *different* unit sample responses.

If the linear system is *locally shift-invariant,* then the shape of the unit sample response does not change much from point to point. An optical lens, for example, is often modeled as a locally shift-invariant system because the point-spread function

changes slowly with increasing off-axis distance [41]. Fig. 2.3 simulates such an effect. If this assumption is valid, then

$$g_{n_1-l_1,\,n_2-l_2}(l_1,l_2) \approx g_{n_1n_2}(l_1,l_2) \tag{4.6}$$

for $-M_1 \le l_1 \le M_1$ and $-M_2 \le l_2 \le M_2$. Eq. (4.5) then becomes

$$y(n_1,n_2) \approx \sum_{l_1=-M_1}^{M_1} \sum_{l_2=-M_2}^{M_2} x(n_1-l_1,\,n_2-l_2)\,g_{n_1n_2}(l_1,l_2). \tag{4.7}$$

In this approximation, the filter coefficients contributing to a specific output point come from a *single* unit sample response.

Finally, if the linear system is *shift-invariant,* then $g_{n_1-l_1,n_2-l_2}(\cdot,\cdot) = g_{00}(\cdot,\cdot) = h_{00}(\cdot,\cdot) = h(\cdot,\cdot)$, and (4.5) reduces to the familiar convolution sum:

$$y(n_1,n_2) = \sum_{l_1=-M_1}^{M_1} \sum_{l_2=-M_2}^{M_2} x(n_1-l_1,\,n_2-l_2)\,h(l_1,l_2). \tag{4.8}$$

4.1.2. Normalization of Filtered Sequences

When filtering digitized sequences, it is desirable to normalize the filtered sequence so that the dynamic range of the output sequence roughly matches that of the input sequence. For linear shift-invariant or locally shift-invariant systems, normalization is performed by one of three equivalent methods: (1) normalizing the unit sample response so that it integrates to unity, (2) dividing the result of each convolution sum by the sum of the unit sample coefficients, or (3) filtering a unity sequence (*i.e.*, a sequence consisting of all 1's) by the unit sample response, and dividing the unnormalized filtered sequence by the result.

For linear shift-variant systems, these three methods are not equivalent. An unbiased normalization requires that each filtered point be divided by the sum of all the coefficients from all the different filters that went into the computation of that point. Method 1 cannot achieve this type of unbiased normalization. Method 2 can be generalized to normalize correctly, but one must carefully keep track of which coefficients actually contribute to the output point, especially when interpolative filtering is performed. Method 3 will also work; when interpolative filtering is performed, the unnormalized sequence is divided by a filtered array of zero-padded unit samples. For shift-invariant interpolation, this is equivalent to polyphase normalization [98].

4.1.3. Adaptive Filter Parameters

Although solid-state display technology is making great progress, it is not yet mature enough to compete with conventional cathode-ray devices. Electron beam displays offer the highest flexibility for the lowest cost. Dynamic focusing of electron beams has been used to compensate for screen curvature, especially in the corners [68]; however, its potential for adaptive image interpolation has not been investigated. Such a scheme is proposed here.

An electron beam is often modeled by a 2-D circular Gaussian current density [56]. Consider a generalized 2-D Gaussian current density of the form

$$B(x,y) = A \cdot \frac{2}{\pi LS} \cdot \exp\left(-2((x'/L)^2 + (y'/S)^2) \right) , \tag{4.9}$$

$$x' = x \cdot \cos\theta - y \cdot \sin\theta , \quad y' = x \cdot \sin\theta + y \cdot \cos\theta ,$$

$$L = 2\sigma_x , \quad S = 2\sigma_y ,$$

where A is the beam *amplitude*, L and S are the long- and short-axis *half-lengths*, and θ is the long-axis *orientation*. These parameters are shown in Fig. 4.3. Writing the Gaussian function in terms of L and S yields a simple rule of thumb: if the

interpolation factor is I, then good isotropic interpolation occurs if $L = S = I$ pixels. This is true because for raster-scanned imagery, a round Gaussian spot will produce a nearly flat-field response when the scan line pitch is approximately 2σ. When $L \neq S$, the beam has an elliptical cross section. The $2/\pi LS$ scaling factor insures that all beam densities integrate to the same total beam current, for constant beam amplitude.

Since the amplitude of the video signal determines A, the important parameters for adaptive 2-D Gaussian filtering are L, S, and θ. In practice, S can be fixed at some optimum value determined by the interpolation factor; therefore, the number of parameters reduces to two: L and θ. Thus, adaptive filtering using 2-D Gaussian beams requires computing L and θ at each point on the output grid.

The analysis in this section deals with adaptive interpolation filters that are Gaussian in shape. This is a restriction imposed by the physics of electron beams. If the interpolation filter could assume *any* optimal shape determined by the local image statistics [99], then the reconstruction could probably be improved; however, the increase in the number of parameters and in the computational complexity

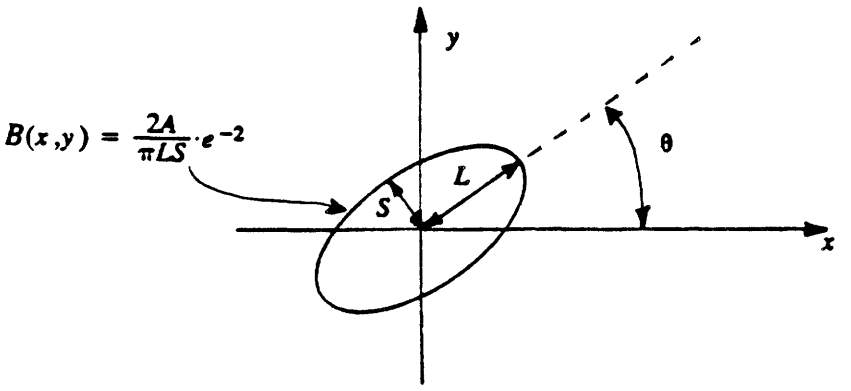


Fig. 4.3. 2-D Gaussian filter parameters.

would render real-time processing impractical.

4.1.4. Finding Edges in Images

Adaptive filtering is beginning to find use in many areas of image processing [100]. In almost all applications, edge information is used to compute or vary the local processing parameters. For instance, Wilson *et al.* [101] used gradient and edge orientation information to design anisotropic nonstationary image estimators, and Ikonomopoulos *et al.* [102] used directional filtering to code the high spatial frequencies of natural images. Similar techniques can be used to control the Gaussian filter shape in adaptive image interpolation. Adaptive Gaussian interpolation in CRT's means that as the electron spot sweeps at constant velocity across the phosphor screen, its size and orientation change in a way that reduces the visibility of sampling structure and aliasing, yet minimizes blur. This is achieved by filtering *along* the principal orientations of lines and edges in images. Therefore, edge reconstruction without "the jaggies" requires knowledge of edge orientation. To distinguish edges from texture and uniform areas, the "strength" of the edge must be measured. These topics are treated below.

4.1.4.1. Determining Edge Orientation

Many common edge-detection algorithms quantize edge orientation to a small number of directions, usually eight or sixteen [103]. Since an elliptical Gaussian beam can be oriented at any angle, better filtering will result if the full range of orientation is used. An algorithm to find edges at *any* angle can be designed quite easily if the image is smooth enough. The problem of determining edge orientation with high precision is most easily attacked by visualizing the array of video samples as a smooth 2-D surface.

Consider a 2-D surface described by the function $f(x,y)$. If the surface is sufficiently smooth, a *normal vector* and *tangent plane* can be associated with each point on it, as shown in Fig. 4.4. The tangent plane at location (x_0,y_0) is defined by the equation [104]

$$f_x(x_0,y_0)\cdot(x-x_0) + f_y(x_0,y_0)\cdot(y-y_0) = f(x,y)-f(x_0,y_0), \qquad (4.10)$$

where $f_x = \partial f/\partial x$ and $f_y = \partial f/\partial y$.

The intersection of the tangent plane with the xy-plane is a straight line. Solving (4.10) for y yields

$$y = -\frac{f_x}{f_y}\cdot x + b, \qquad (4.11)$$

where b is the y-intercept. The slope of this line is the *orientation* at location (x_0,y_0). The angle θ between this line and the x-axis is given by [105]

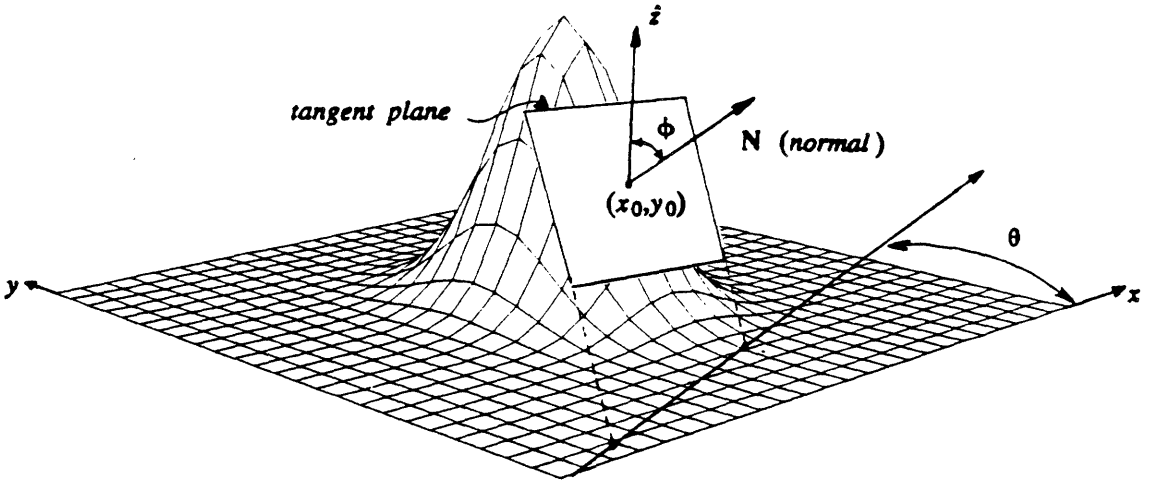


Fig. 4.4. A 2-D surface with normal vector and tangent plane.

$$\theta(x,y) = \arctan\left(-\frac{f_x}{f_y}\right). \tag{4.12}$$

In practice, the continuous function $f(x,y)$ is sampled, producing the discrete function $f(n_1,n_2)$. A discrete approximation to the orientation formula in (4.12) that centers the orientation estimate over the sample being examined is

$$\theta(n_1,n_2) = \arctan\left(\frac{f(n_1-1,n_2) - f(n_1+1,n_2)}{f(n_1,n_2+1) - f(n_1,n_2-1)}\right). \tag{4.13}$$

This 3×3 neighborhood computation requires 2 subtractions, a division, and an arctan computation for each point in the image. If the image is digitized to 8 bits per sample, then the division and arctan computations can be replaced by a single look-up operation; therefore, to determine the orientation at each point requires only *two parallel subtractions and a single table look-up operation,* which can be done at video rates. Another advantage of this algorithm is that θ can assume *any value* between $-\pi$ and π. Since a 2-D Gaussian is symmetric with respect to the origin, only the absolute value of θ is needed to specify beam orientation.

4.1.4.2. Determining Edge Strength

A number of metrics can be used to determine the steepness, or strength, of edges in an image [81]. Considered here are simple metrics that can be computed over a 3×3 region in real time. Desirable characteristics of an edge metric are: (1) high response at an edge, (2) low response in uniform areas or texture, and (3) relative insensitivity to noise [106]. Some simple edge metrics are listed below:

4.1.4.2.1. Gradient Magnitude

$$\beta_1(x,y) = \sqrt{f_x^2 + f_y^2} \tag{4.14}$$

The gradient magnitude gives a highly localized response proportional to edge steepness. However, sometimes it is difficult to distinguish between edges and noise or between edges and texture, especially when the signal-to-noise ratio (SNR) is very poor.

4.1.4.2.2. Slope Average

$$\beta_2(x,y) = \frac{|f_x| + |f_y|}{2} \tag{4.15}$$

The slope average is similar in performance to the gradient magnitude, but is simpler and faster to compute. Digital implementation requires only two subtractions, one addition, and one binary shifting operation for the division by 2. This can be computed in real time.

4.1.4.2.3. Normal Angle

$$\beta_3(x,y) = \phi(x,y) = \arccos\left(\frac{1}{\sqrt{f_x^2 + f_y^2 + 1}}\right) \tag{4.16}$$

The *normal angle,* shown as ϕ in Fig. 4.4, is the angle between the normal vector and the z-axis. This metric gives a high response to edges and lines of *any* amplitude; however, it is more sensitive to noise than the previous two metrics.

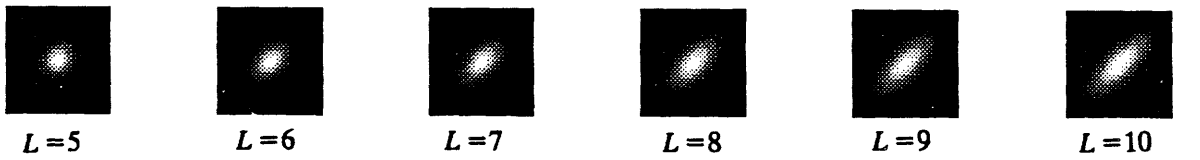
4.1.4.3. The Smoothness Assumption

The above algorithms assume that the image is a smooth 2-D surface. When is the smoothness assumption valid? It is certainly valid for those input images that have been adequately prefiltered prior to subsampling. In tube-type video cameras, the lens and the read beam spot usually provide enough spatial prefiltering to avoid severe aliasing in most natural imagery. However, some aliasing is present in the raster image [69], and if the display is receiving a highly resolved image from a high-definition camera or from a computer frame store, then the smoothness assumption may be invalid, and the edge-finding algorithms may give poor results. If samples from a single video field are used to determine edge information, vertical aliasing will be more of a problem. In this case, it may be better to lowpass filter the signal before edge processing.

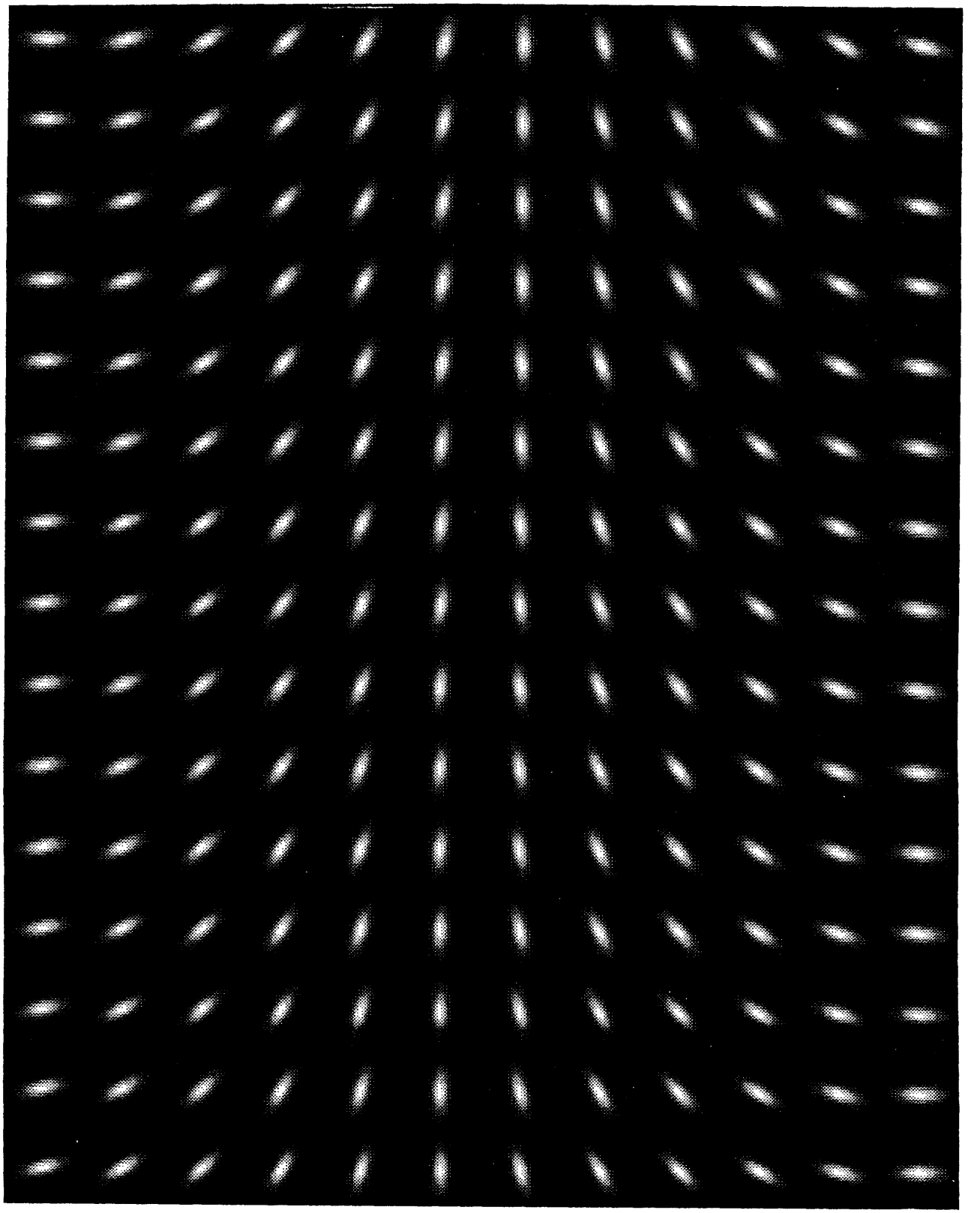
The smoothness assumption also influences the choice of filtering algorithm for discrete simulation. If the edges vary smoothly from point to point, then the locally shift-invariant filtering algorithm in (4.7) can be used. However, for aliased imagery, edge orientation may vary erratically over a small region, and the general filtering algorithm in (4.5) must be used for better results.

4.1.5. Experimental Results

Experiments in adaptive image interpolation were carried out on both a PDP 11/45 and a VAX 11/785 computer. Programs were written to implement the shift-variant and locally shift-invariant filtering algorithms described by (4.5) and (4.7). Images were filtered, subsampled 4:1, zero-padded 4:1, and then adaptively filtered with interpolators whose shape and orientation varied with local edge characteristics. The filters were picked from a large set of 2-D Gaussians described by (4.9). Since the interpolation factor was 4, the short-axis half-length was fixed at $S=4$. Six values of L and 180 values of θ were used to form a matrix consisting of 1080 different interpolation filters, each of size 25×25. This allows good modeling of a Gaussian electron beam having a wide range of cross-sectional orientations and eccentricities. Shown in Fig. 4.5 is a subset of the filter matrix.



(a) Six Gaussian filters with $S=4$ and $\theta=45°$



(b) Gaussian filter subset with $S=4$, $L=10$ and θ ranging from 0 to 180° in 1° increments.

Fig. 4.5. Subset of a 2-D Gaussian filter matrix.

4.1.5.1. Adaptive Interpolation in Two Dimensions

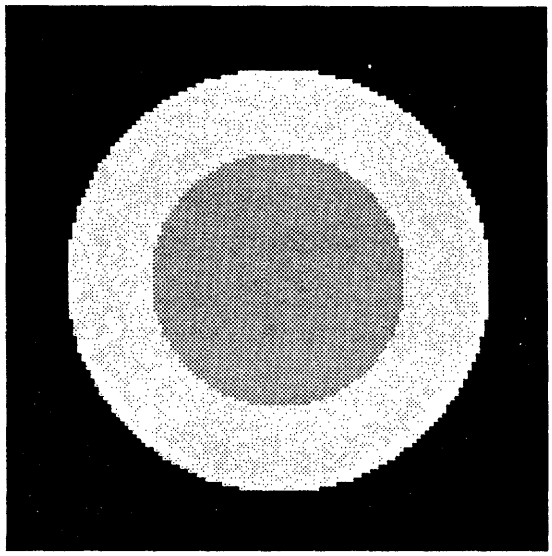
The first set of experiments deals with images that are subsampled and interpolated in *both* dimensions. The locally shift-invariant algorithm in (4.7) was used for these images. A later set of experiments simulates the tube-type scanning found in conventional cameras and displays. The shift-variant algorithm of (4.5) was used for these images.

4.1.5.1.1. Circular Test Pattern

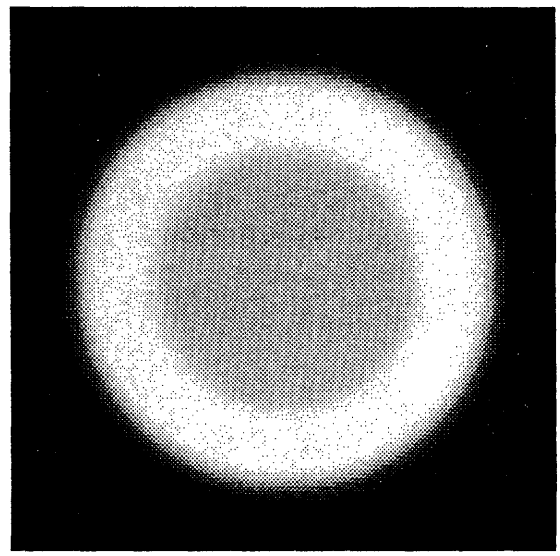
The first set of experiments deals with a synthetically generated test pattern consisting of two concentric circular patches. The original is shown in Fig. 4.6(a). This pattern was chosen because it contains edges at all orientations, and the amplitude of the outer circular edge is greater than that of the inner one. Fig. 4.6(b) is the original image filtered by a 19× 19-pt. Gaussian with 95% of its energy within a radius of $\rho = \pi/4$ in the frequency domain. Fig. 4.6(c) is the 4:1 subsampled version of 4.6(b), and Fig. 4.6(d) is the 4:1 zero-padded version of 4.6(c). The latter supplies the output grid for interpolation.

In Fig. 4.7 are shown four non-adaptive filtering methods: (a) boxcar, (b) bilinear, (c) truncated $\sin x/x$, (d) truncated Gaussian with 95% energy within $\pi/4$. Comparing these images with the ideal image in Fig. 4.6(b) reveals that Gaussian interpolation performs best, even though sampling structure is visible and the circular edges are a bit jagged.

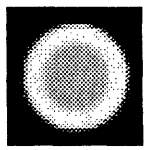
Edge information derived from Fig. 4.6(b) is shown in Fig. 4.8. In practice, only the information in 4.6(c) is available at the receiver; however, the purpose of this experiment is to compare the quality of various edge algorithms. Fig. 4.8(a) displays orientation information using the formula in (4.13). Intensity corresponds to edge direction (θ); black is 0 radians, medium gray is $\pi/2$ radians, and white is π radians. Notice how smoothly the intensity varies along the two circular bands. The large intensity changes at the top and bottom of the bands correspond to phase jumps of π radians in edge direction. Figs. 4.8(b), (c), and (d) display the gradient magnitude, slope average, and normal angle edge strength metrics.



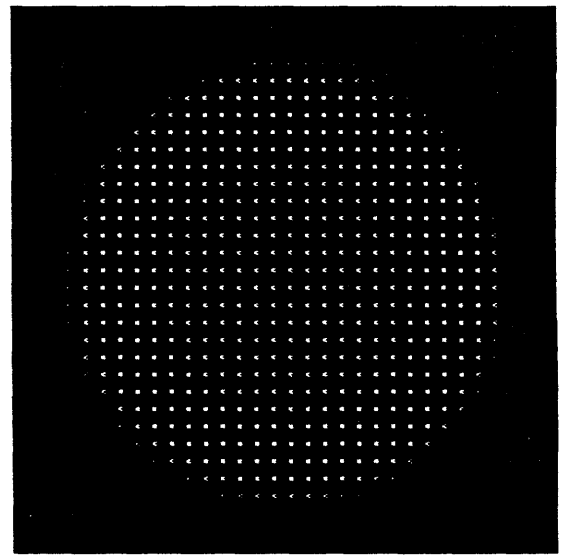
(a) Original (128 × 128)



(b) Fig. 4.6(a) filtered to $\rho=\pi/4$

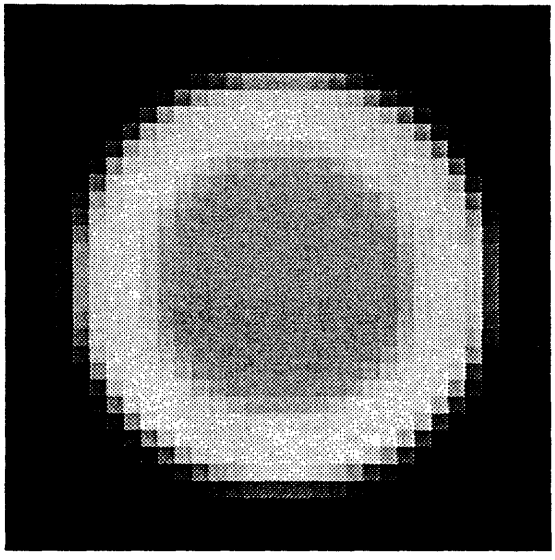


(c) Fig. 4.6(b) subsampled 4:1

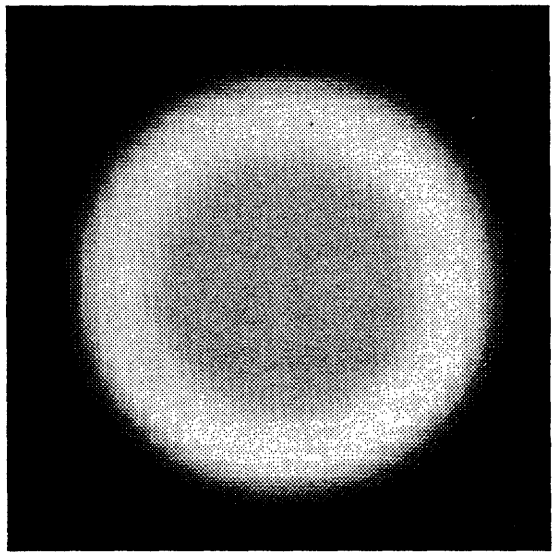


(d) Fig. 4.6(c) zero-padded 4:1

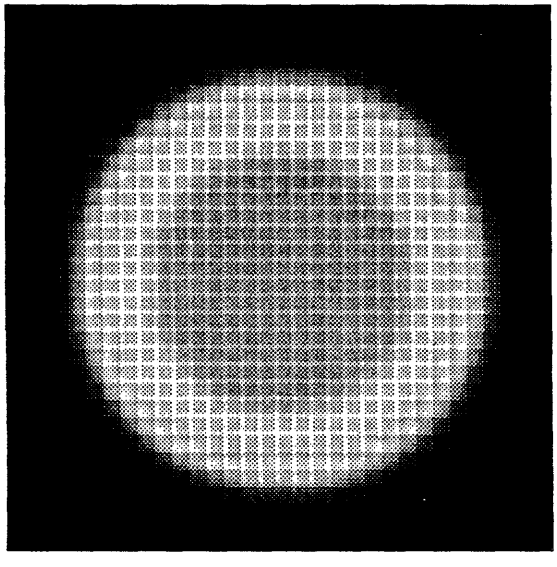
Fig. 4.6. Circle test pattern.



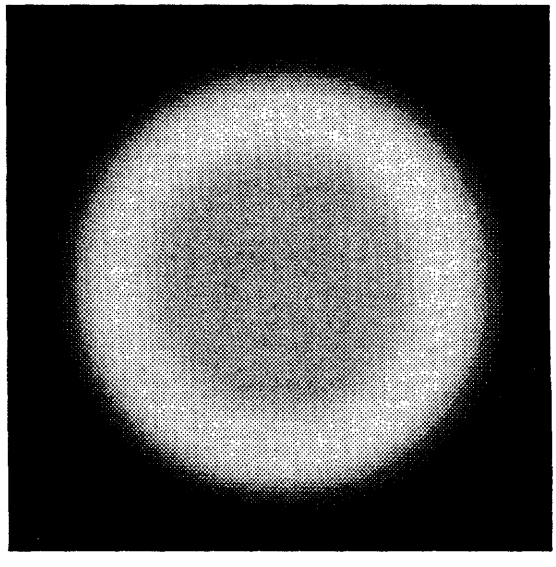
(a) Boxcar filter



(b) Bilinear filter

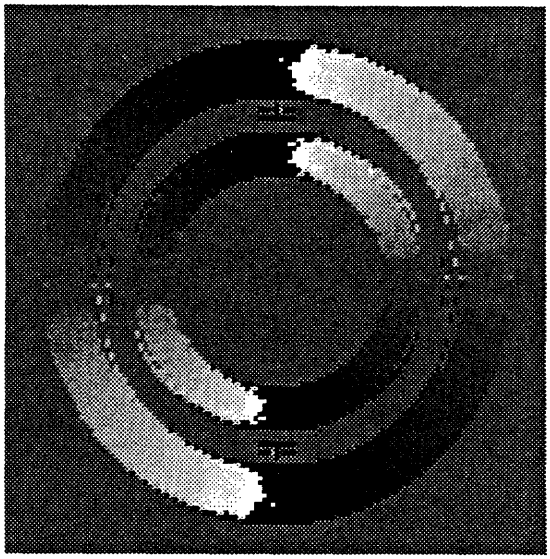


(c) Truncated $\sin x/x$ filter

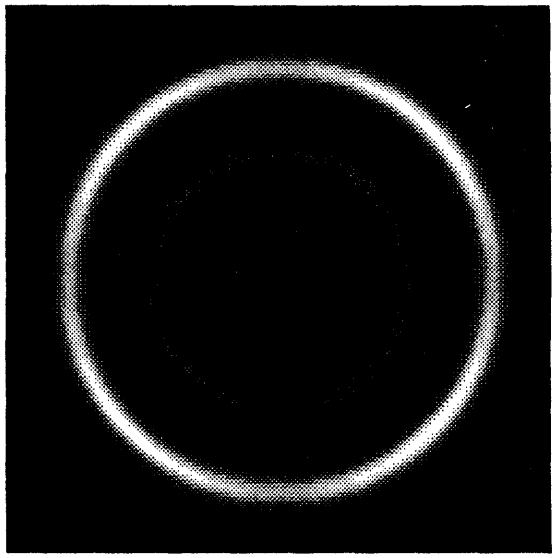


(d) Truncated 19×19 Gaussian filter

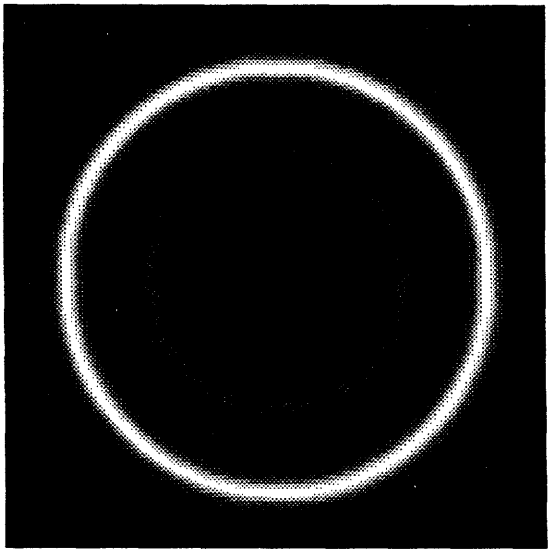
Fig. 4.7. Isotropically filtered versions of Fig. 4.6(d).



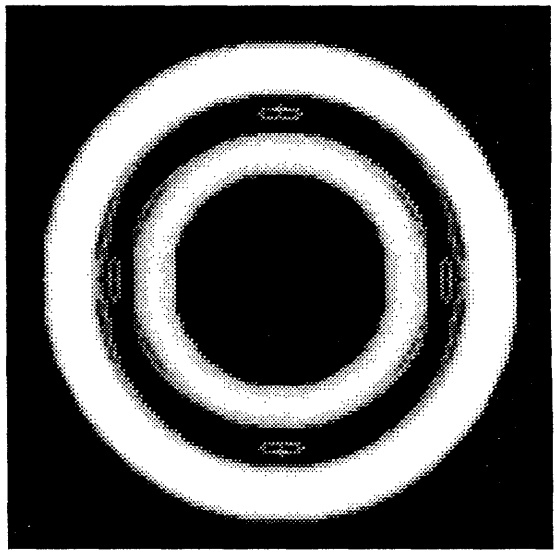
(a) Edge orientation



(b) Gradient magnitude



(c) Slope average



(d) Normal angle

Fig. 4.8. Edge information derived from Fig. 4.6(b).

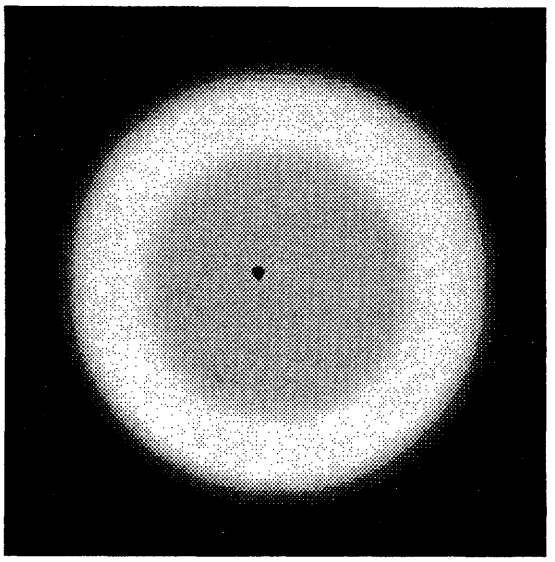
Fig. 4.9 displays adaptive interpolation methods using different edge strength metrics. 4.9(a) is equivalent to 4.6(b), the filtered original. Figs. 4.9(b), (c), and (d) are adaptively filtered versions of Fig. 4.6(d) in which the L parameter is linearly proportional to the gradient magnitude, slope average, and normal angle, respectively. In all cases, θ is determined using the edge orientation formula in (4.13). Figs. 4.9(b) and (c) are virtually identical and show good filtering at the edges; 4.9(d), however, shows a poor filtering job at the edges. All exhibit some sampling structure because the S parameter is too small.

The experiment in Fig. 4.10 is identical to that of Fig. 4.9 except that it operated on an original with 10 dB of white Gaussian noise added before prefiltering. The orientation, gradient magnitude, and slope average are insensitive to noise; the normal angle metric appears quite sensitive.

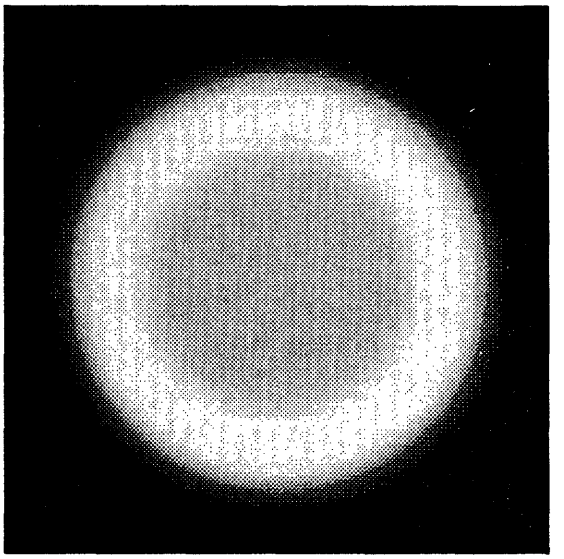
4.1.5.1.2. Cameraman Image

Some experiments were repeated with a natural test image. Figs. 4.11, 4.12 and 4.13 have the same parameters as Figs. 4.6, 4.7, and 4.8. Fig. 4.14 is an experiment that uses the simple orientation formula in (4.13) to determine θ, and the simple slope average formula in (4.15) to determine L. The slope average metric is used to control beam eccentricity for the following reasons: in uniform or slowly varying areas, there are no strong edges, so a circular spot is the best choice because it is unbiased. An edge, however, is reproduced without "the jaggies" by a highly eccentric filter with its long axis parallel to the edge. The slope average metric is simple to compute, and also varies *smoothly* away from edges, implying a smooth change in filter eccentricity.

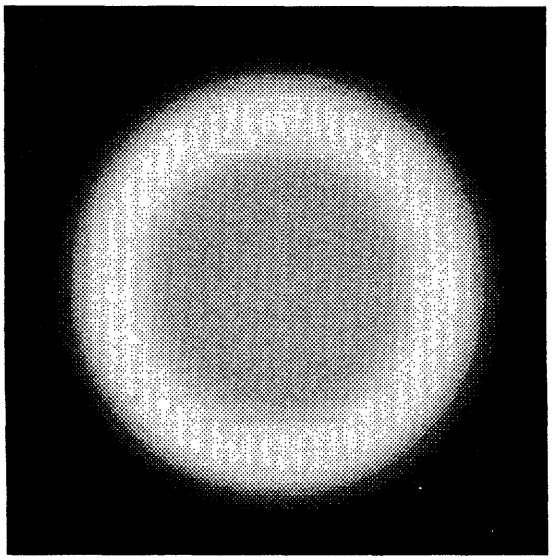
All the images in Fig. 4.14 are adaptively filtered versions of Fig. 4.11(d). Fig. 4.14(a) used edge information from the prefiltered original, Fig. 4.11(b). In a sense, this represents the ideal case, because the prefiltered image is not available at the receiver. When edge information is derived from the subsampled image, Fig. 4.11(c), and then bilinearly interpolated to the size of the output grid, Fig. 4.14(b) results. Note the artifacts along some edges. Another possibility is to derive edge



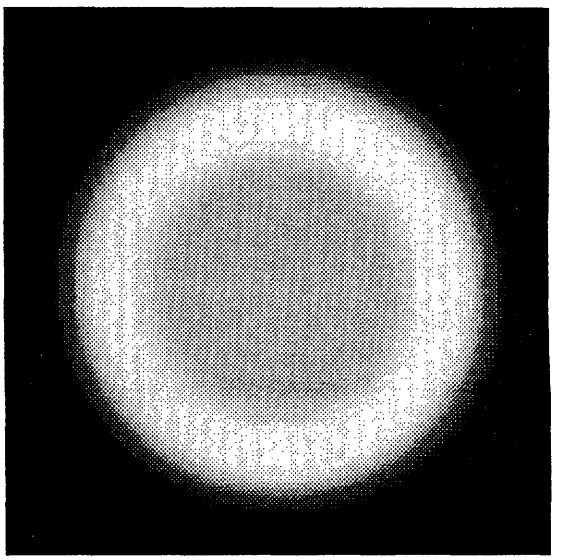
(a) Fig. 4.6(b)



(b) Using gradient magnitude

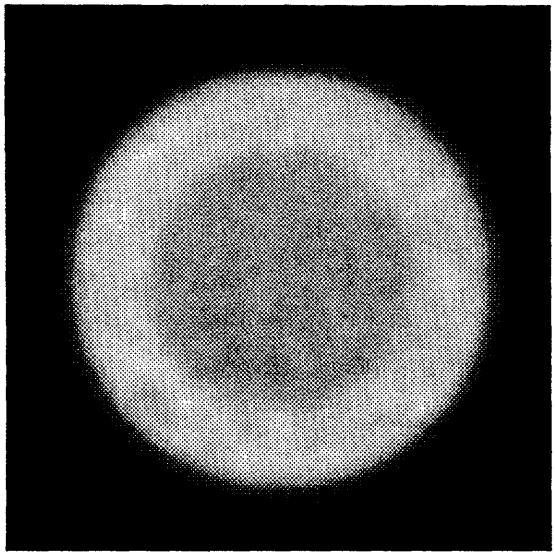


(c) Using slope average

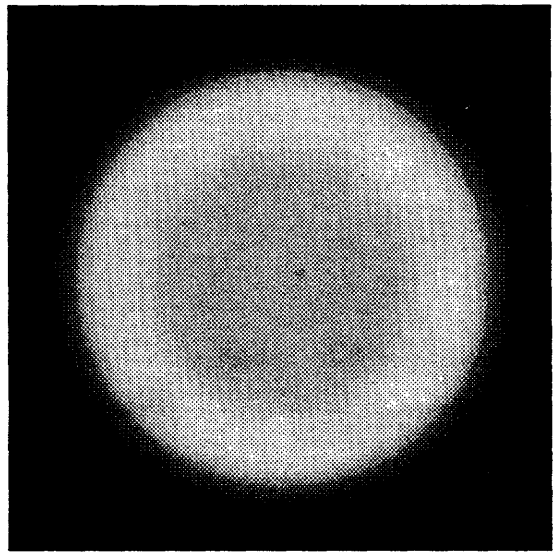


(d) Using normal angle

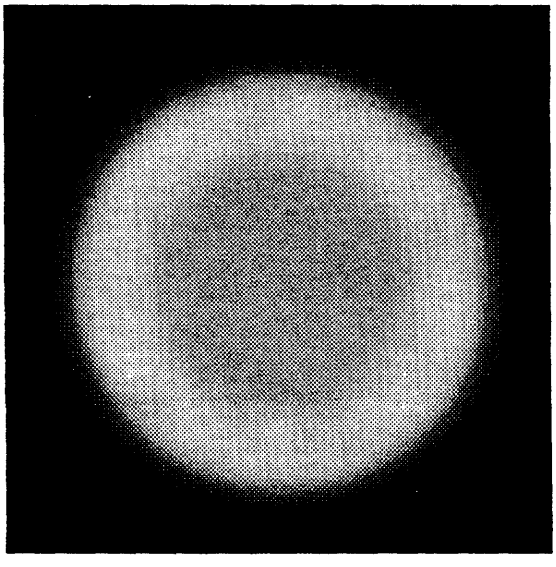
Fig. 4.9. Adaptively filtered versions of Fig. 4.6(d) using various edge strength metrics.



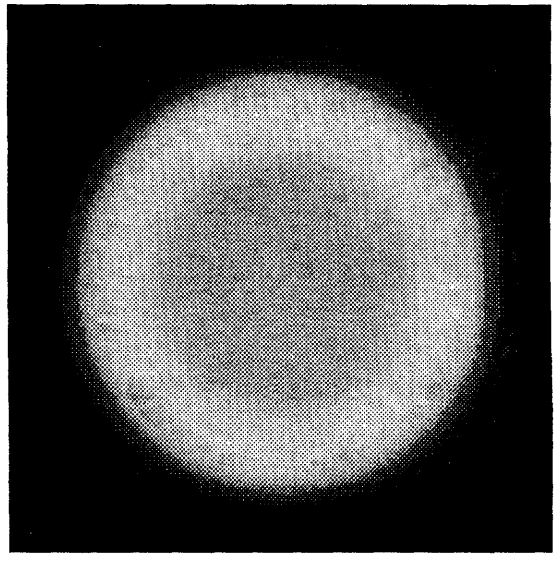
(a) Filtered version of Fig. 4.6(a) with 10 dB SNR



(b) Using gradient magnitude



(c) Using slope average



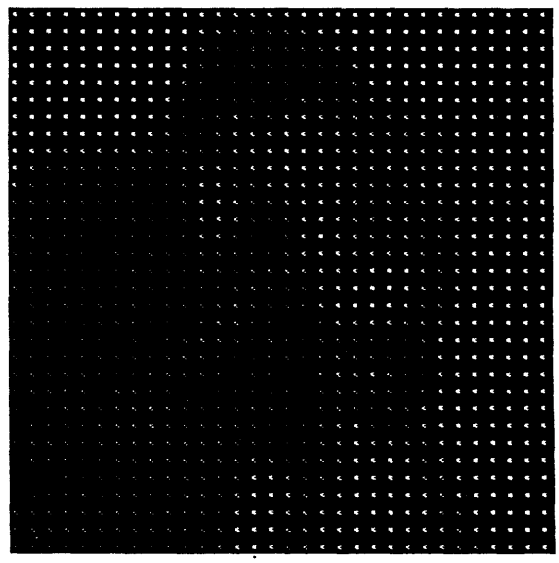
(d) Using normal angle

Fig. 4.10. Adaptively filtered versions of noisy original using various edge strength metrics.

(a) Original (128 × 128)

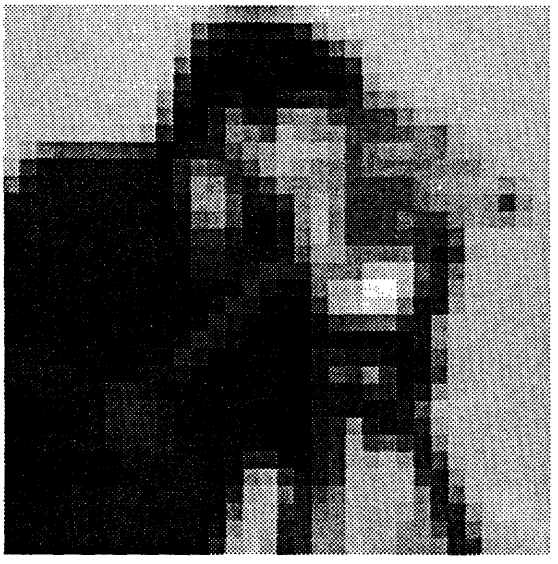
(b) Fig. 4.11(a) filtered to $\rho = \pi/4$

(c) Fig. 4.11(b) subsampled 4:1



(d) Fig. 4.11(c) zero-padded 4:1

Fig. 4.11. Cameraman test image.



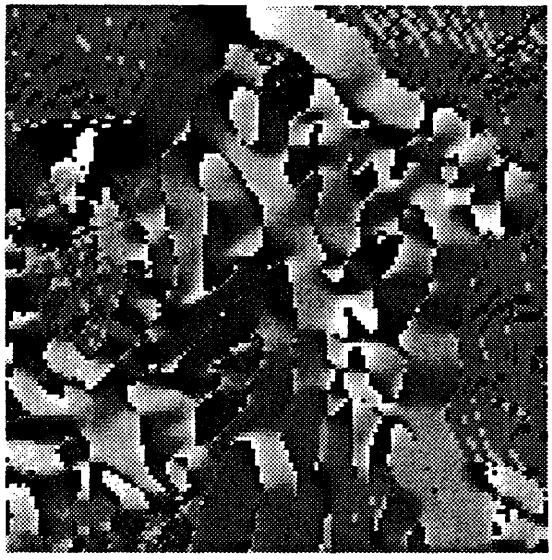
(a) Boxcar filter

(b) Bilinear filter

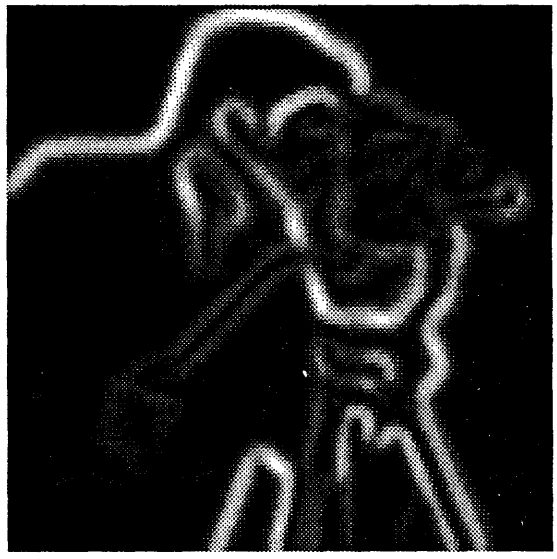
(c) Truncated $\sin x/x$ filter

(d) Truncated 19×19 Gaussian filter

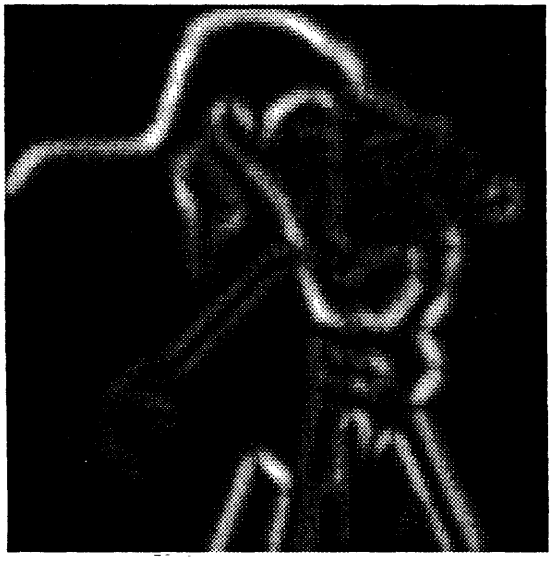
Fig. 4.12. Isotropically filtered versions of Fig. 4.11(d).



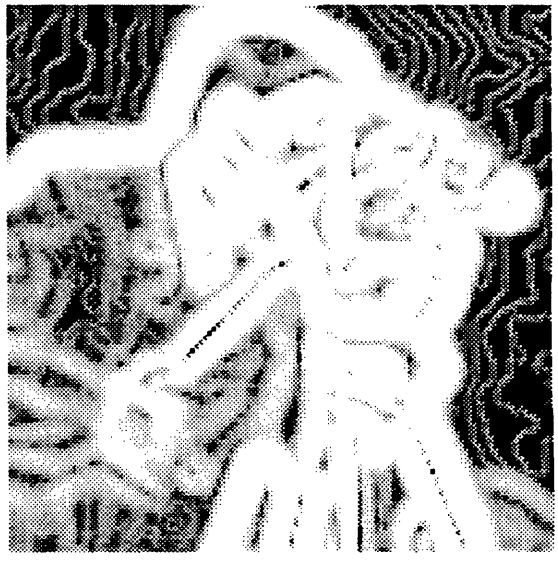
(a) Edge orientation



(b) Gradient magnitude



(c) Slope average



(d) Normal angle

Fig. 4.13. Edge information derived from Fig. 4.11(b).

(a) Derived from Fig. 4.11(b)

(b) Derived from Fig. 4.11(c), then bilinearly interpolated

(c) Derived from Fig. 4.12(b)

(d) Derived from Fig. 4.12(d)

Fig. 4.14. Adaptively filtered versions of Fig. 4.11(d) using edge metrics derived in various ways.

information from a non-adaptively interpolated version of the received signal. Edge information derived from a bilinearly-interpolated version of the received signal, Fig. 4.12(b), results in Fig. 4.14(c). Edge information derived from a Gaussian-interpolated version of the received signal, Fig. 4.12(d), results in Fig. 4.14(d). In all of these adaptively filtered images, edges are rendered without "the jaggies", and much of the detail found in a good non-adaptively filtered image, Fig. 4.12(d), is retained. These preliminary results indicate that adaptive filtering performs better when edge information is derived from an interpolated signal rather than interpolated from the subsampled signal.

4.1.5.2. Adaptive Interpolation in One Dimension

This section discusses how adaptive filtering techniques can be applied to existing digital TV receivers. Storage of only three lines of video is needed to implement the simplest edge-finding algorithms in (4.13) and (4.15). The only requirement is that the video signal be sampled at a rate that preserves the aspect ratio of the image; this insures that correct edge orientation information is applied to the output grid. The L and θ values computed in real time at each point in the digitized image are used to control the shape and tilt of the electron beam dynamically.

Figs. 4.15 and 4.16 show the effect of beam eccentricity on image quality when θ is known exactly. The originals are shown in (a) and are chosen to show clearly the effect of spatial aliasing, which shows up as jagged edges and moire patterns. Aliasing occurs because the originals are subsampled vertically 4:1 without prefiltering. The zero-padded video signals are shown in Figs. 4.15(b) and 4.16(b). In (c) are shown the result of isotropic filtering with $L=S=4$. Moire patterns and jagged edges are clearly visible. In Figs. 4.15(d)-(i) and 4.16(d)-(i), the long-axis half-length, L, is incremented from 5 to 10 pixels, and unnormalized adaptive filtering is performed. The jagged edges slowly disappear as L increases. The test pattern in Fig. 4.15 is sensitive to subsampling phase [66]; a different subsampling phase may improve or degrade the vertical moire pattern that is still visible in Fig. 4.15(i).

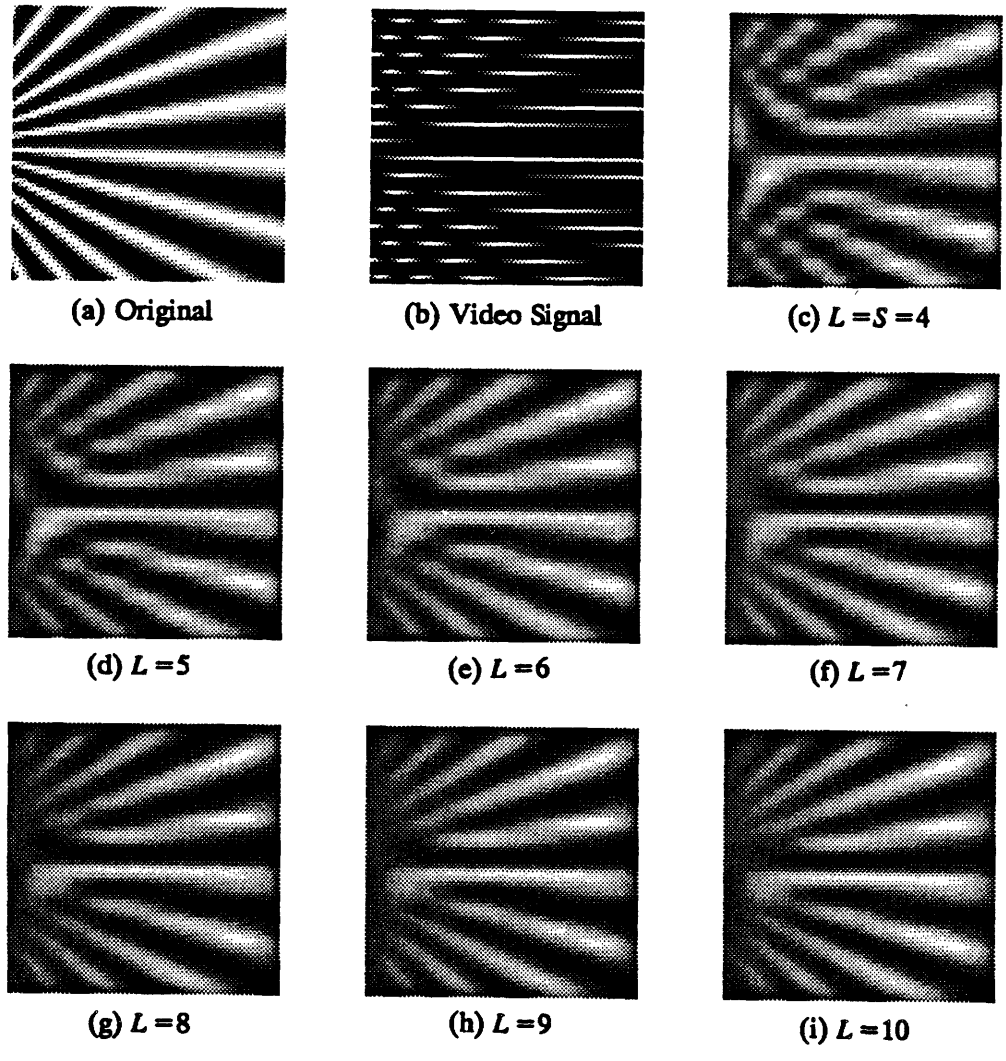


Fig. 4.15. Effect of beam eccentricity on image quality when edge orientation is known exactly.

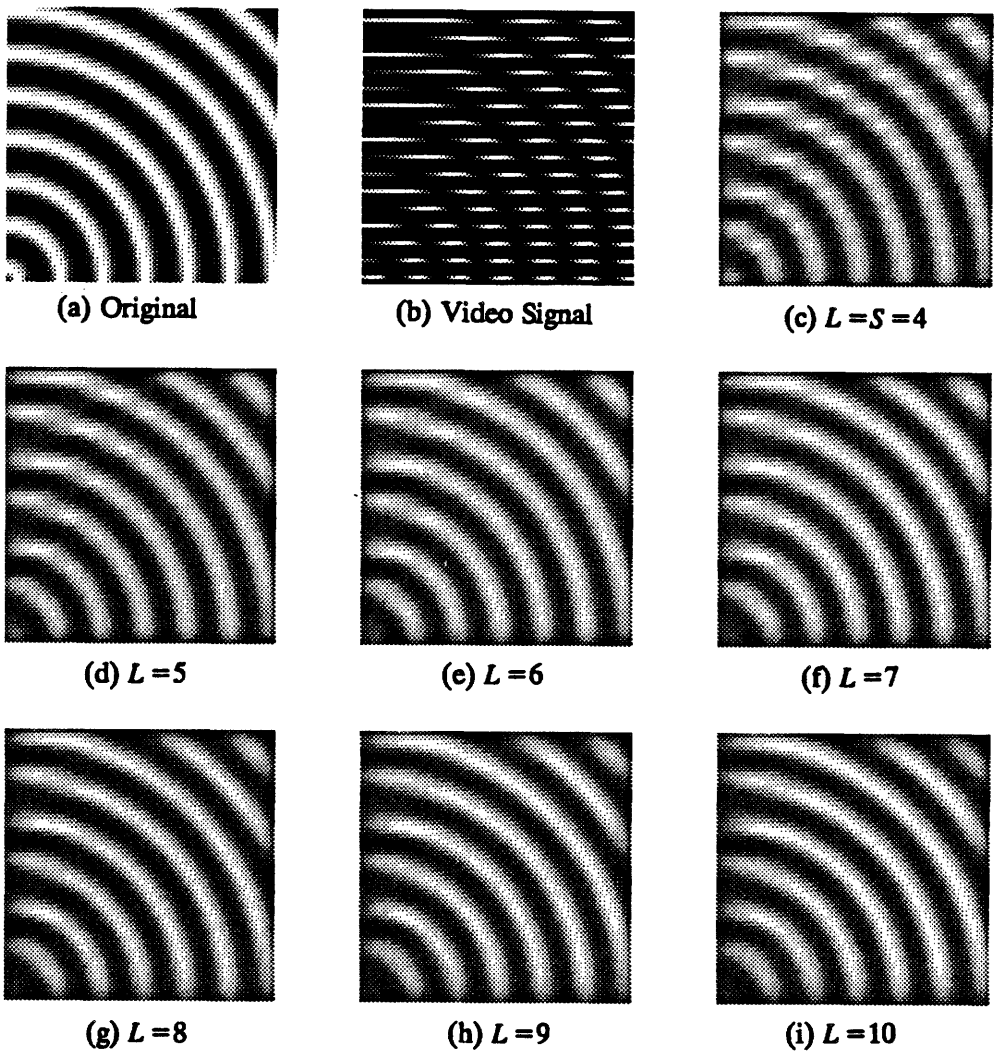


Fig. 4.16. Effect of beam eccentricity on image quality when edge orientation is known exactly.

The effect of normalization on image quality is investigated in Fig. 4.17. The left column shows unnormalized images; the right, images normalized by dividing by a filtered zero-padded array of unit samples. The normalized images show improved contrast and fewer objectionable artifacts. In (a), the edge information is known exactly; in (b), edge information is derived from the original image; in (c), edge information is derived from the subsampled image and then linearly interpolated to the dimensions of the output grid; in (d), edge information is derived after linearly interpolating the subsampled image. The random brightness variations in (b), (c), and (d) are probably due to edge orientation errors derived from the aliased signal. Unfortunately, there is no perfect way to normalize the electron beam filtering in a real CRT. The interpolated images would look more like those on the left.

Adaptive Gaussian interpolation can also be used to synthesize new lines for high-resolution displays, although the method described in the next section appears to give better results. First, the video samples are adaptively interpolated in a frame buffer using a *sharpened* elliptical Gaussian filter. Sharpening is performed along the short axis of the filter so that edge contrast is improved. Normalization is performed digitally, and the interpolated image is then scanned with a small circular beam onto a high-resolution display. A simulation of adaptive line interpolation is shown in Fig. 4.18. Fig. 4.18(a) shows 64 simulated scan lines of a natural image displayed with a circular Gaussian spot. Fig. 4.18(b) shows the result of adaptive line interpolation. The 64 video lines are adaptively interpolated by a factor of 4, and 256 lines are displayed with a small circular spot on a simulated high line rate monitor or on a multi-beam display [107]. Edge orientation is determined from a filtered version of the original signal. The adaptive filter is a Gaussian with $L=10$ and $S=4$, sharpened along its short axis to reduce intersymbol interference [47]. No aliasing or scan lines appear in the adaptively interpolated image, and it is sharper than the image displayed without interpolation. If adaptive line interpolation can be performed in real time, it can produce sharper images with decreased scan line visibility.

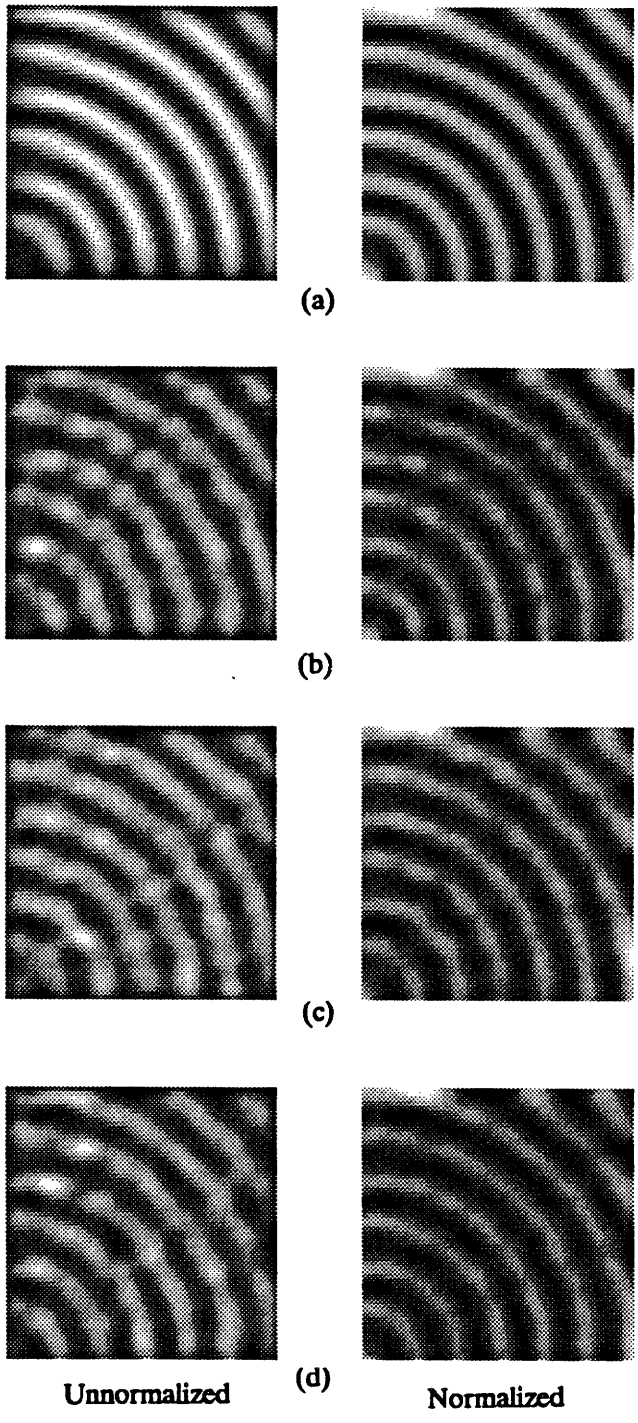


Fig. 4.17. Effect of normalization on image quality when edge orientation is (a) known exactly, (b) derived from original, (c) derived from subsampled image, then linearly interpolated, and (d) derived from linearly interpolated subsampled image.

(a)

(b)

Fig. 4.18. Simulation of adaptive line interpolation. (a) 64 simulated scan lines, displayed on a standard CRT. (b) 4:1 adaptive line interpolation of video signal used in (a), displayed on a high-resolution CRT.

4.2. Adaptive Digital Line Interpolation

In this section, the second method of adaptive image interpolation is described. This digital technique interpolates contours using a block-matching algorithm, whereas the previous method used an analog beam to adaptively filter the raster image. Another difference is the magnitude of intersymbol interference produced by each. The analog beam filter may interfere with signal values on adjacent lines; the digital filter does not disturb the values of the original samples and hence has zero intersymbol interference. Despite these differences, there is an important similarity. Both methods respond to edges and filter along the edges, thereby reducing aliasing artifacts.

A promising technique for enhanced-definition television (EDTV) is digital line interpolation at the display [94]. The simplest non-adaptive methods, line replication and line averaging, produce noticeable aliasing artifacts. Motion-adaptive methods [108, 109, 110] produce better results, but aliasing along moving contours is still observed. Described here is a method of adaptive intrafield line interpolation that greatly reduces aliasing and performs better than non-adaptive techniques. It can be used at video rates to enhance the performance of motion-adaptive interpolators. This technique can also be used as a one- or two-dimensional interpolator for digitized imagery.

4.2.1. Nonadaptive Methods

One proposal for EDTV is conversion from 60 fields per second to 60 frames per second at the display. This requires 2:1 line interpolation in each field, as shown in Fig. 4.19. The simplest hardware interpolators use non-adaptive methods. Roberts [111] was among the first to report improved performance using a motion-adaptive line interpolation technique: intrafield or interfield averaging is chosen based on local motion measurements. For objects in motion, intrafield averaging will almost always be chosen, and aliased moving edges may appear. If the intrafield interpolation is adaptive, however, improved performance can be expected.

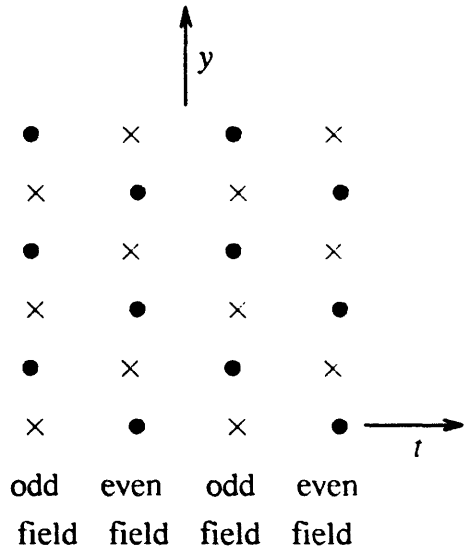


Fig. 4.19. Line interpolation for 2:1 interlaced signals. Circles indicate transmitted lines. Crosses indicate interpolated lines.

In intrafield interpolation, the interpolated pixels are synthesized from pixels in lines above and below. Fig. 4.20 shows the spatial sampling geometry for 2:1 line interpolation. Samples from two adjacent lines in the same field, y and z, are used to determine samples in the synthesized line, x. Let x_n be the nth sample in line x. In practice, no more than two original lines are used to synthesize intermediate lines. Filtering more intrafield lines may produce better results but would require more video storage and increased computation.

4.2.1.1. Line Replication

The simplest method of line interpolation is line replication. No arithmetic operations are involved, and only one video line must be stored. The algorithm is

$$x_n = y_n \ . \tag{4.17}$$

Although uniform areas are rendered perfectly, sloping contours exhibit aliasing, the main drawback of this method. A frequency domain analysis shows why aliasing occurs. A single digitized field from an interlaced frame has a spectrum with baseband replications at $\omega_y = \pm\pi$. Vertical line replication is equivalent to multiplying the spectrum by a function approximating $\sin\omega_y/\omega_y$, whose zero crossings are at the vertical replication frequencies. Although filtering occurs at high vertical frequencies, it is not enough to eliminate the lower-frequency aliasing components. The result is jagged edges and moire patterns.

4.2.1.2. Line Averaging

Another non-adaptive method is line averaging. It requires only two arithmetic operations per interpolated point:

$$x_n = \frac{y_n + z_n}{2} \ . \tag{4.18}$$

Line averaging performs perfectly in uniform areas of the picture, but it too produces jagged edges. However, its performance is usually better than line replication for most imagery. In the frequency domain, vertical line averaging is equivalent to filtering with a triangular weighting function, whose spectrum is approximately $(\sin\omega_y/\omega_y)^2$. Multiplying this function by the replicated spectrum of a single field attenuates all vertical frequency components more strongly than the $\sin\omega_y/\omega_y$-like function of line replication. Thus, aliasing components are reduced, but so is valid vertical detail in the alias-free regions.

4.2.2. Motion-Adaptive Method

In still portions of the picture, temporal (inter-field) averaging is better because it preserves vertical detail. In moving portions, spatial (intra-field) averaging is better because it preserves temporal detail. These observations have led to the development of motion-adaptive line interpolation techniques. Although they differ in certain details, the main idea is to to switch between inter- and intra-field averages based on local motion measurements. Fukinuki *et al.* [96] used the following interpolation algorithm in a prototype motion-adaptive high-definition scan converter:

$$x_n = k \cdot \frac{a_n + b_n}{2} + (1-k) \cdot \frac{y_n + z_n}{2} , \qquad (4.19)$$

where a_n and b_n are samples in the same spatial position from the previous and following field, and $0 \leq k \leq 1$ is a movement coefficient determined from vertical and temporal differences.

A natural extension to this algorithm is to introduce another level of adaptation. Better interpolation can be realized by interpolating contours either spatially or temporally. The following section discusses the first of these, spatial contour interpolation.

4.2.3. Novel Block-Matching Method

If edge contours can be interpolated accurately between lines, then aliasing, a major source of image degradation, can be eliminated or greatly reduced. The idea of contour interpolation is not new; it was first proposed as a bandwidth compression scheme in 1961 by Gabor and Hill [112]. They simulated the technique with a photomechanical model. Since then, similar methods have been proposed for video coding [113], image interpolation [114], and color enhancement [115]. However, previous efforts were either too complex to work at video rates or were not optimized for video line interpolation.

The method of contour interpolation proposed here is designed specifically for 2:1 line interpolation of monochrome images. It works by matching small blocks of pixels in lines y and z and finding the position of best match over a certain range. This position determines which pixels to use for interpolation.

Relevant parameters are B, the block size, and R, the range of pixels (maximum block offset) used in the search. These are shown in Fig. 4.20. At zero offset, the blocks are centered above and below x_n. To avoid biasing, B should be odd: $B=2M+1$, where M is a nonnegative integer. Blocks move one pixel at a time until the entire range is covered. Moving the blocks in opposite directions along lines y and z insures that x_n always lies at the midpoint of a line connecting the middle of the two blocks. Block differences are produced at each offset:

$$BD_i = \sum_{m=-M}^{M} | y_{n+m+i} - z_{n+m-i} | , \quad -R \le i \le R , \tag{4.20}$$

where BD_i is the block difference at offset i. The smaller the block difference, the better the match. The smallest block difference determines the best offset, i^*:

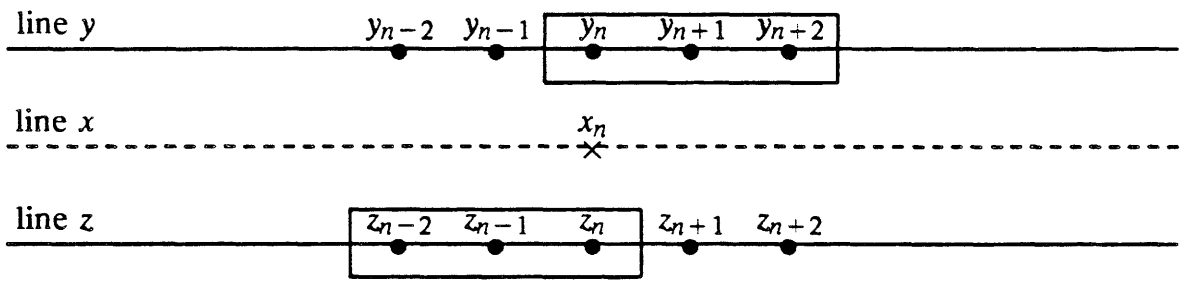


Fig. 4.20. Spatial sampling geometry for 2:1 line interpolation. In this example, $B=3$, $R=1$, and the current offset is $i=1$.

$$\min_i \{BD_i\} = BD_{i^*} \ . \tag{4.21}$$

Finally, this optimal offset determines the pixels used in the interpolation:

$$x_n = \frac{y_{n+i^*} + z_{n-i^*}}{2} \ . \tag{4.22}$$

As in line averaging, x_n is an average of two pixel values in lines y and z. However, these pixels are not necessarily the ones directly above and below x_n. Instead, they are found at the intersection of the line of highest correlation passing through x_n. For instance, an edge at $+45^\circ$ (on the sampling grid) would produce an optimal offset of $i^* = 1$, so that $x_n = (y_{n+1} + z_{n-1})/2$.

Computing block differences for all offset values requires $(2R+1)B$ subtractions and $(2R+1)(B-1)$ additions. $2R+1$ block differences must be searched for a minimum, and the final computation requires 1 addition and 1 division.

4.2.3.1. Experimental Results

Figs. 4.21 and 4.23 show the improved performance of adaptive line interpolation. Fig. 4.21 is a natural image and Fig. 4.23 is a computer-generated test pattern designed to show aliasing artifacts. Corresponding images in the two figures are processed with identical parameters. In (a) is shown the original frame (256×256) consisting of two interlaced fields. The even field is discarded, and the remaining images are the result of 2:1 line interpolation using the *odd field only*. Line replication is shown in (c) and line averaging is shown in (d). Vertical aliasing is visible in both of these interpolated images. Adaptive line interpolation using the method described here is shown in (b). The block size is 7 and the range is 1. In the natural image, no aliasing is visible. There are few noticeable artifacts, and the overall quality is better. In the test pattern, aliasing does occur, but at a higher spatial frequency. Extending the algorithm to subpixel accuracy is discussed in a later section; it improves the performance but also increases the computation rate.

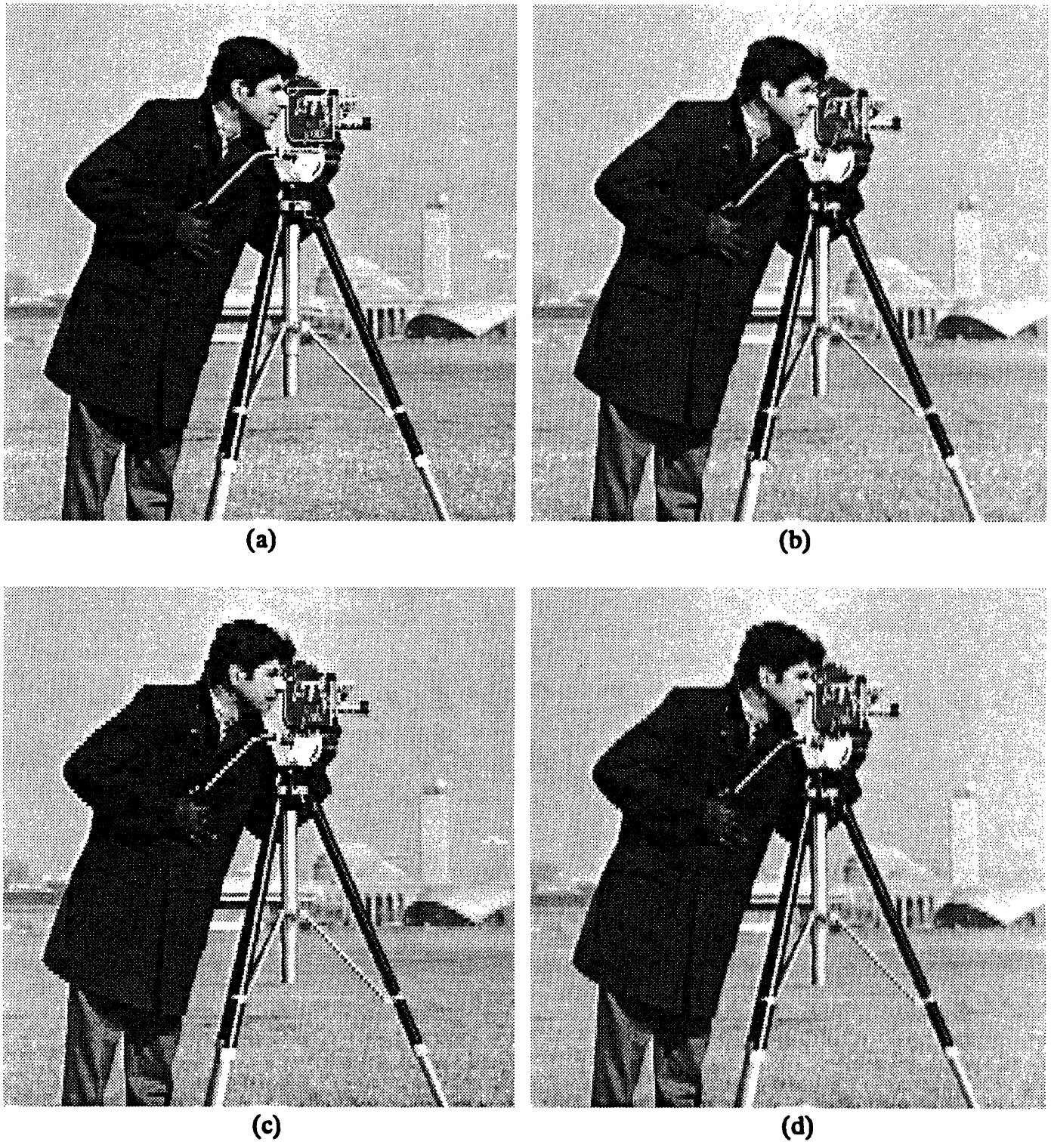
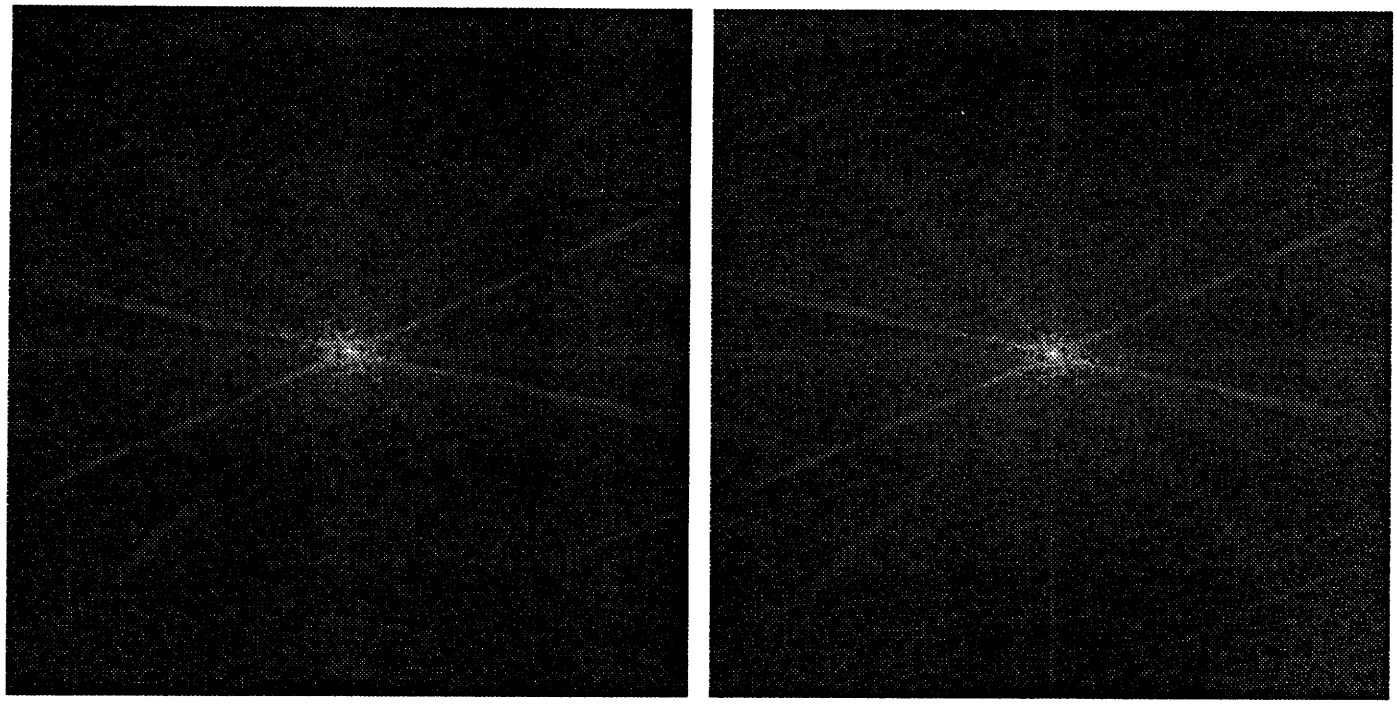
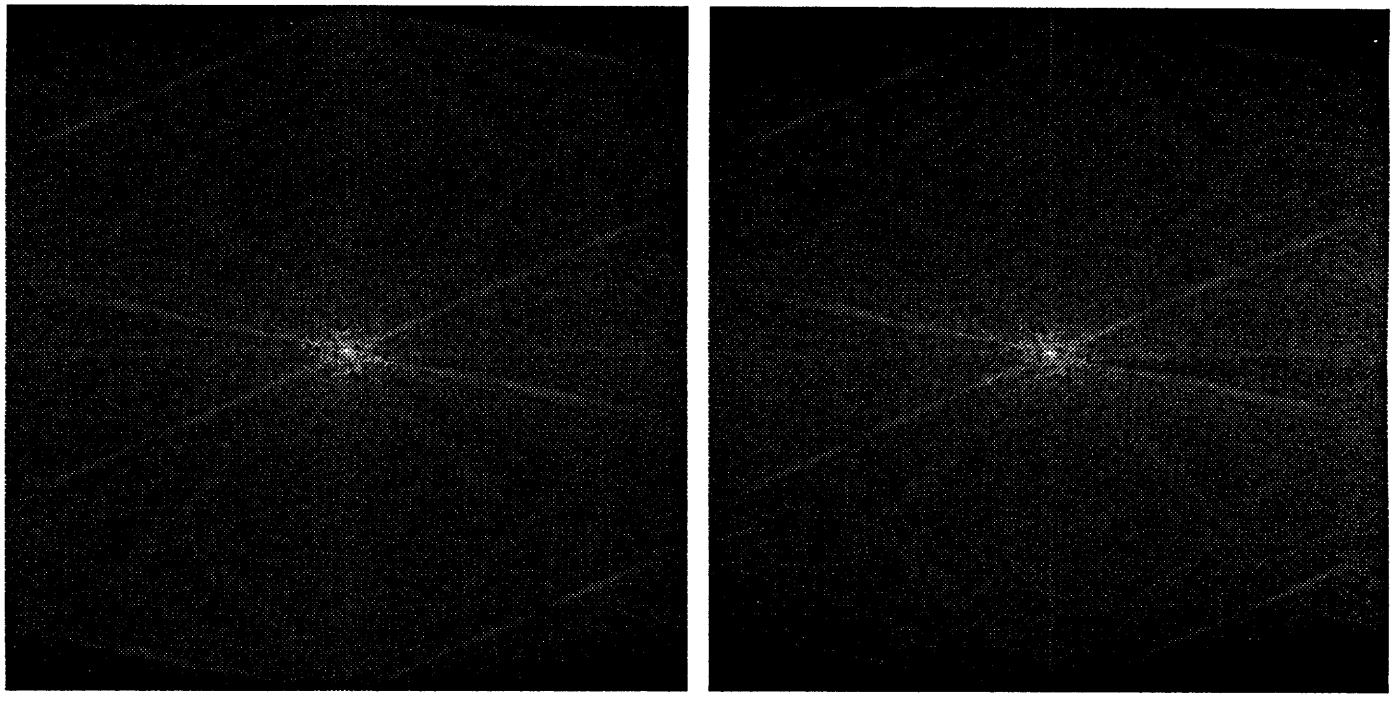


Fig. 4.21 2:1 line interpolation from odd field, natural image. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.



(a) (b)



(c) (d)

Fig. 4.22 Spectra of images in Fig. 4.21. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.

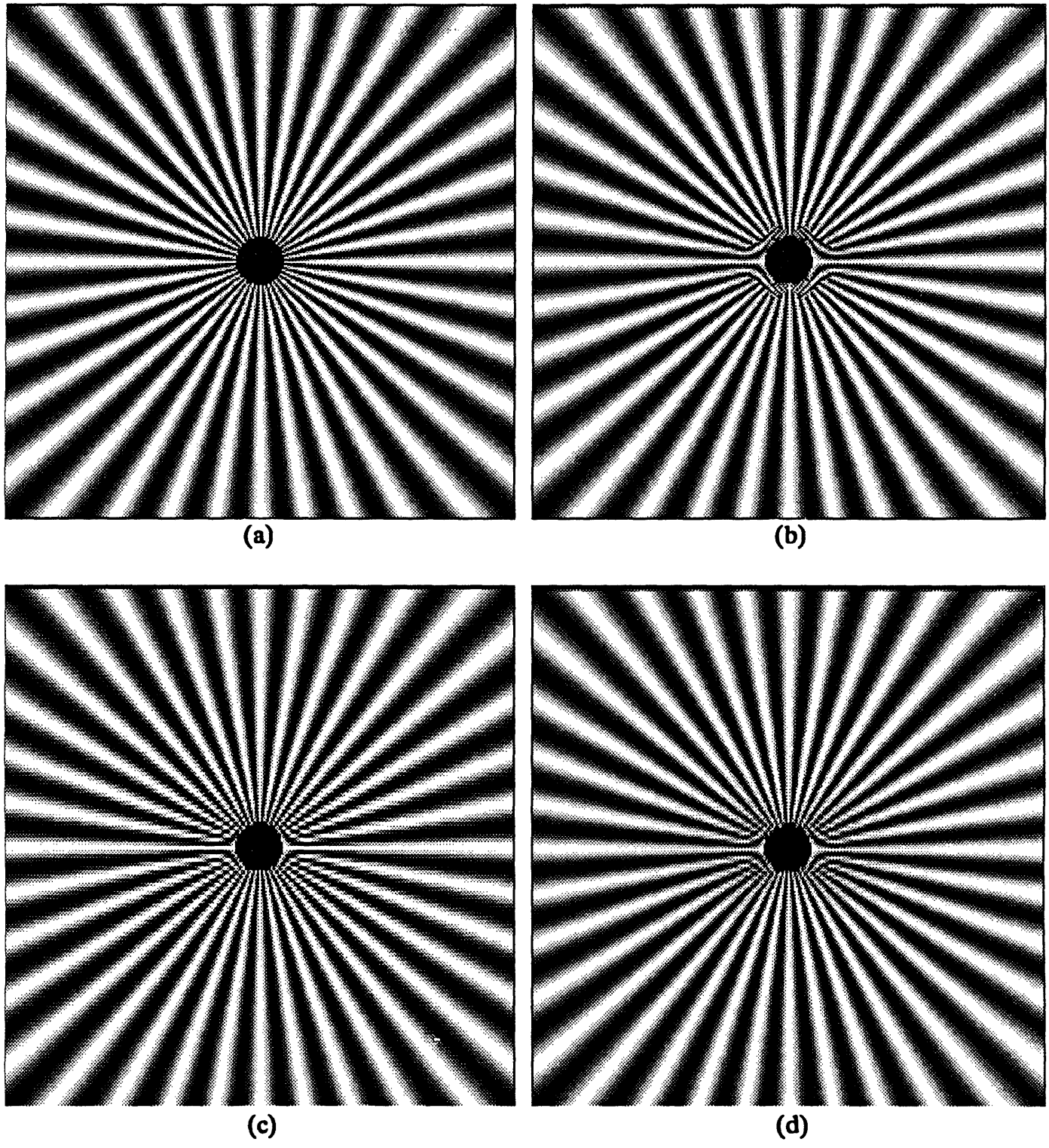


Fig. 4.23 2:1 line interpolation from odd field, test pattern. (a) Original (256×256). (b) Adaptive block-matching method. (c) Nonadaptive line replication. (d) Nonadaptive line averaging.

4.2.3.2. Effect of Block Parameters on Image Quality

The block size and range of search affect the quality of adaptive line interpolation. Both were varied and the rms error between the interpolated image and the original was computed for the cameraman image. Fig. 4.24 shows the results. The rms error predicts the subjective quality fairly accurately, especially for changes in R.

The smallest errors are produced at the smallest range value, $R=1$. This is because larger ranges fool the algorithm into aligning similar detail that is not spatially coherent. In the cameraman image, large range values produce annoying artifacts at the intersection of sharp edges, as in the lower portion of the tripod. Large range values, however, do perform better in certain situations, such as interpolating slightly sloped contours.

The block size has less of an effect on image quality. The largest improvement occurs when B is increased from 3 to 5. A larger block size is needed to insure that an edge is actually present within the block. However, if the block size is too large, vertical detail that is not spatially coherent may fool the matching algorithm. At $B=5$, a few artifacts remained in both the natural and synthetic images. However, at $B=7$ and $R=1$, most artifacts disappear.

4.2.3.3. Frequency Domain Interpretation

Some insight into the improved performance of this algorithm can be found in the frequency domain. In Fig. 4.22 is shown the log magnitude Fourier spectra of the cameraman images of Fig. 4.21. The spectrum of the original frame is shown in (a). Discarding the even field without filtering causes the spectrum to replicate at the vertical frequencies of $\pm\pi$, which corresponds to the top and bottom of each spectrum. The alias components are evident in (c) and (d), even after multiplying by $\sin\omega_y/\omega_y$- and $(\sin\omega_y/\omega_y)^2$-like functions. However, almost no aliasing is visible in (b). Furthermore, much of the original vertical detail is preserved.

The alias-free spectrum of Fig. 4.22(b) is actually the result of local alias-removal processing. Any edge in the original image that is not completely vertical

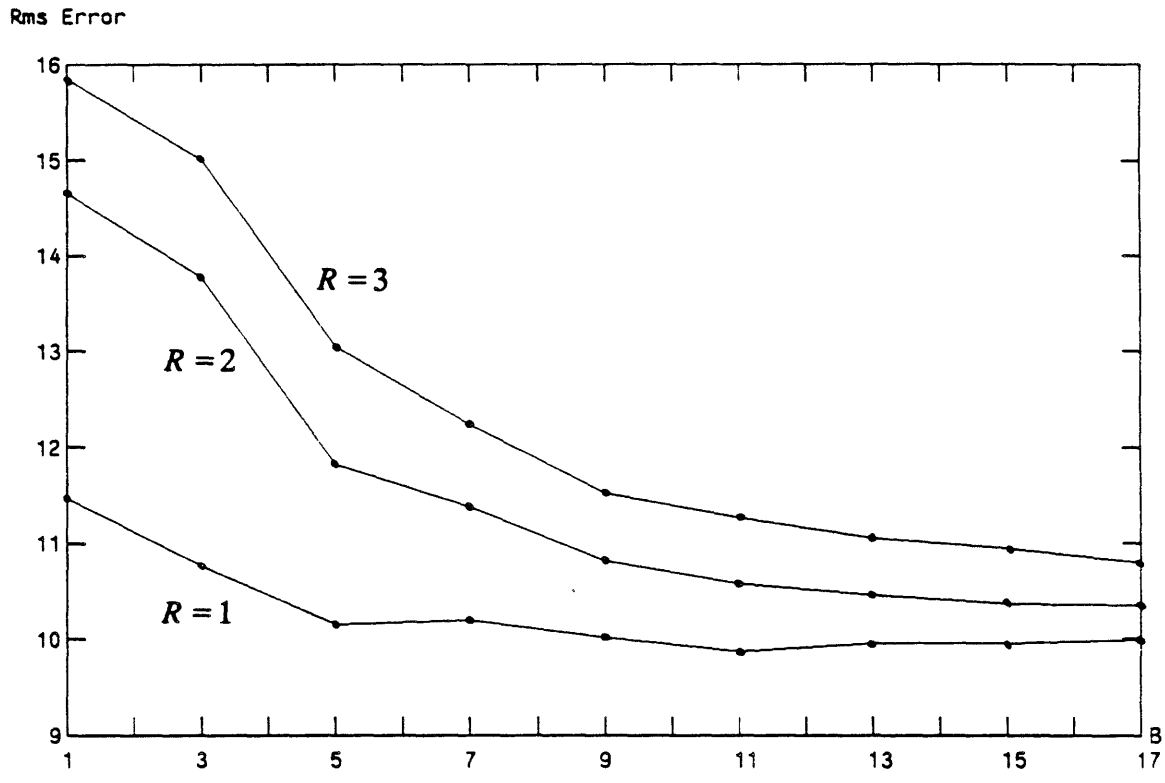


Fig. 4.24. Rms error as a function of block size (B) and range of search (R).

will introduce an alias component when the even field is discarded. Of course, any high-frequency vertical detail is lost at this point. In natural images, however, a contour usually extends over several lines, providing enough information to interpolate the contour. This is equivalent to filtering *along* the edge, thereby reducing the aliasing component otherwise introduced. A non-adaptive filter can remove or reduce aliasing in a certain region or along a specific direction; however, an adaptive filter can theoretically remove or reduce aliasing along *any* direction. The net effect is an image free from aliasing artifacts.

4.2.3.4. Extending the Algorithm to Subpixel Accuracy

In the block-matching algorithm just described, the blocks move in increments of whole pixels. Because of this, contours are interpolated at fixed angles of 90°,

$\pm 45°$, $\pm 22.5°$, etc., whether or not contours exist at these angles. The detection and interpolation of contours can be made more accurate by "filling in" each line with intermediate pixel values and moving the blocks in increments of subpixels. For instance, accuracy to half a pixel can be realized by interpolating a new value between adjacent horizontal pixels, and then moving the blocks in units of half pixels.

The type of horizontal interpolation affects the quality, but linear interpolation is simple and produces adequate results. The improvement in performance is shown in Fig. 4.25. The original is shown in (a); it is a cropped version (64×64) of the center of the test pattern shown in Fig. 4.23(a). When the even field is discarded, only the odd field remains, as shown in (b). This image is aliased, and it is easy to see why local processing is unable to recover the original image completely. Nevertheless, some algorithms perform better than others. In (c) is shown line replication and in (d) is shown line averaging. The right column shows adaptively interpolated images at various levels of subpixel accuracy. From (e) to (h), the level varies from whole- to half- to third- to quarter-pixel accuracy. In all cases, $B = 7$ and $R = 1$ subpixels. The visual quality improves substantially from whole- to half-pixel accuracy, but less so at finer levels. A larger range value produces better results in the corners of the image, but introduces artifacts near the center. A possible improvement is to make the range adaptive to local detail.

4.3. Summary

The results reported here suggest that adaptive filtering can improve picture quality by reducing sampling structure visibility and aliasing while retaining important image detail. Analog beam shaping was first considered. This method exploits knowledge of edge orientation and strength. A simple algorithm was described that determines edge orientation with high precision, and works well even in the presence of noise. Three measures of edge strength were proposed. The simplest, slope averaging, provides a method for smoothly varying the filter eccentricity across an edge. It also behaves well in the presence of noise. Experiments with adaptive Gaussian interpolation of natural imagery suggest that edge-finding

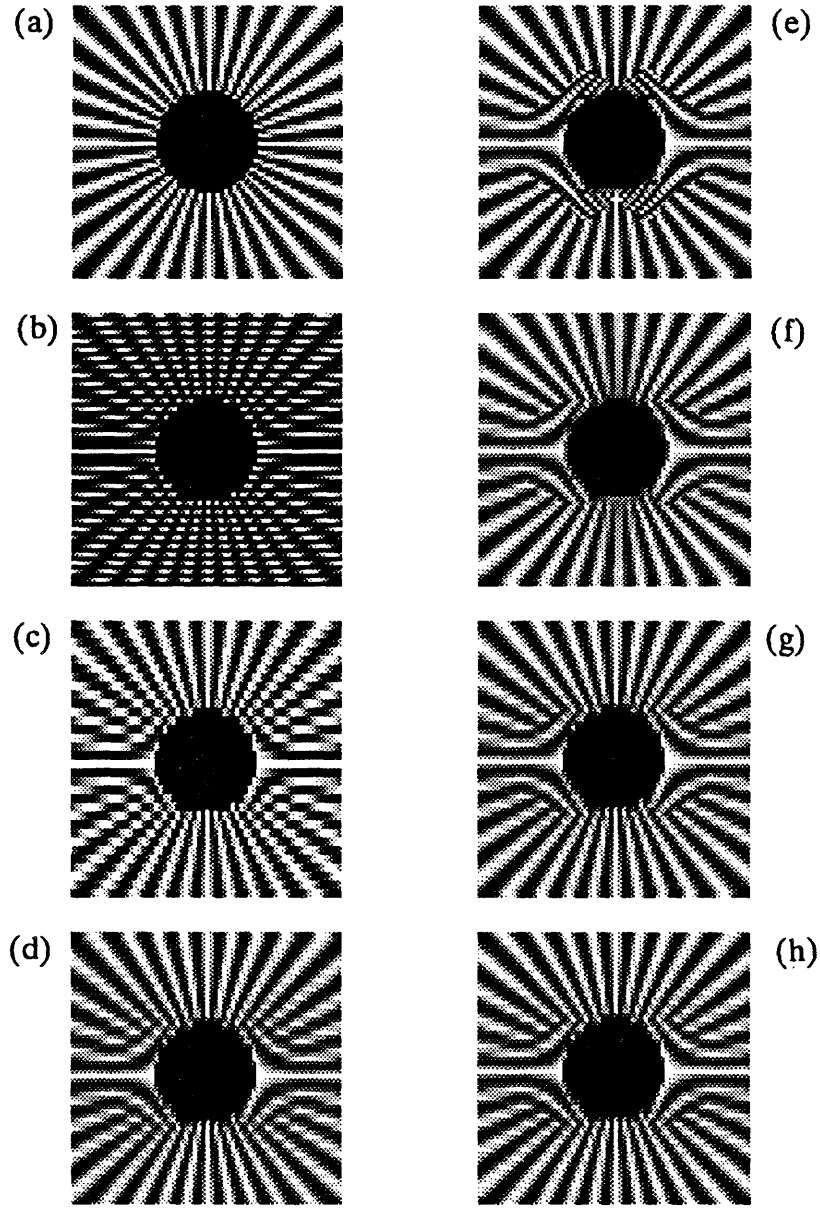


Fig. 4.25 Performance of block-matching algorithm at various levels of subpixel accuracy. (a) Original (64×64). (b) Odd field of (a). (c) 2:1 line replication. (d) 2:1 line averaging. Adaptive block matching ($B = 7$, $R = 1$) at accuracy levels of (e) whole pixel, (f) half pixel, (g) one-third pixel, (h) one-quarter pixel.

algorithms may work better if edges are located *after* bilinear interpolation of the received signal. A method of adaptive line interpolation using sharpened Gaussian filters was described.

A new block-matching method was proposed for adaptive 2:1 line interpolation. This technique can produce a high-quality frame of video from each field; thus, 60 progressively scanned frames per second can replace 30 interlaced frames. The parameters used in this simulation ($B = 7$, $R = 1$) require 21 subtractions, 19 additions, 1 division, and 2 comparisons per interpolated point. Exploiting parallelism, this computation is feasible at real-time rates and could be performed using a single IC chip. Extending the algorithm to subpixel accuracy can improve the performance. Further research should be undertaken to determine optimum parameters for typical video signals.

Chapter 5

TVSIM - A TV Simulation Program

5.0. Introduction

In this chapter, the operation of TVSIM, a TV simulation program, is described. The program is written in C and runs on a PDP 11 or a VAX computer. Because of main memory constraints, only smaller pictures can be processed on the PDP 11. UNIX manual pages for TVSIM and other television simulation programs are given in Appendix A.

TVSIM is designed as a tool for the analysis and design of monochrome baseband TV systems. Programmed for generality, TVSIM has an extensive set of parameters under user control. The chapter concludes with several TVSIM examples.

5.1. Operation of TVSIM

Shown in Fig. 5.1 is a block diagram of the TVSIM program. This is the most general configuration of the standard system components. Each component is a separate program that can run independently of the main TVSIM program. Thus, other system configurations can be designed by rearranging the standard components or by interfacing new components. Although the standard components must occur in the order shown, the user may eliminate any undesired ones. The *crop* and *lens* programs (shown in the dotted box) are used only if required.

TVSIM processes a single digitized monochrome image to produce a single "snapshot" of a simulated monochrome display. The camera operates in storage-erase mode and the display operates in write-store mode, as described in Chapter 2. The dimensions of the input image are arbitrary. Of course, larger sizes allow

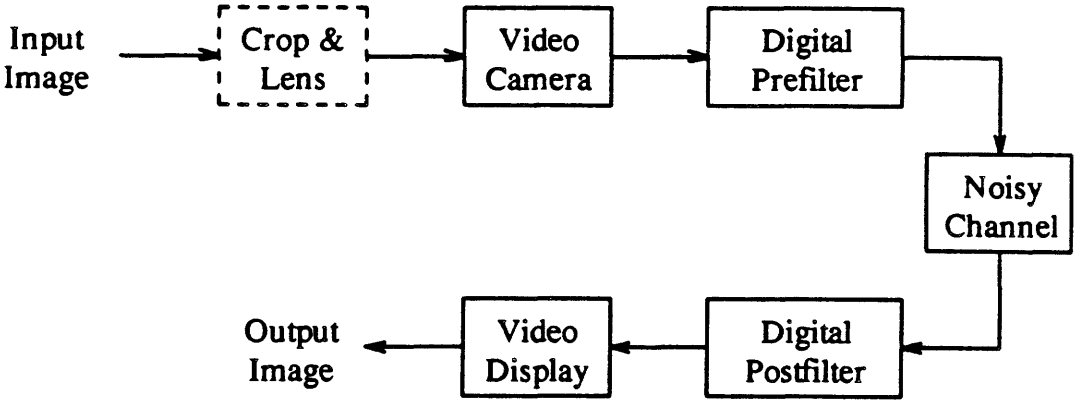


Fig. 5.1. Block diagram of TVSIM program.

simulations of systems of higher definition, but require more processing time.

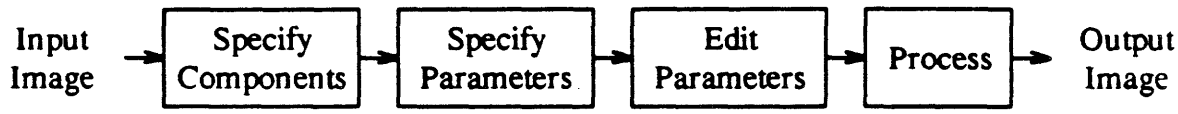
5.1.1. Modes of Operation

TVSIM runs in three modes: design, analysis, and rerun. These are shown schematically in Fig. 5.2 and are explained below.

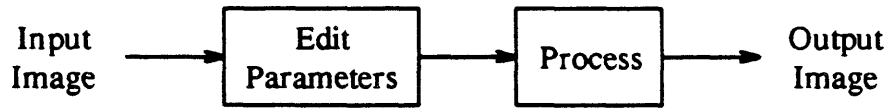
Design mode is the default mode, and is entered by typing the command

tvsim *pic*

where *pic* is the name of the input image datfile. Datfiles are UNIX directories consisting of a descriptor file and a data file, and are described in more detail in Appendix B. If a camera lens is specified, then *pic* may be interpreted as the luminance distribution in the object plane of the camera's optical system. If no lens is specified, then *pic* may be interpreted as the illuminance distribution in the focal plane of the camera's target. Denote the dimensions of *pic* by $H_{INPUT} \times V_{INPUT}$. In design mode, one begins by specifying the TV components to be included in the



(a) Design mode.



(b) Analysis mode.

(c) Rerun mode.

Fig. 5.2. TVSIM modes of operation.

simulation. Then the parameters for each component are interactively specified. After this, one may edit the parameter list to correct any errors. The TVSIM parameters are converted to datfile keywords, appended onto the descriptor portion of *pic*, and processing begins.

Analysis, or *edit, mode* is entered by typing the command

tvsim *pic* -e [*pars*]

where *pic* is the name of an input image datfile and *pars* is an optional ordinary file containing a predefined set of TVSIM parameters. If *pars* is not specified, parameters are taken from the descriptor portion of *pic*, which implies that *pic* should have been run through TVSIM at least once in design mode. After the parameters are

edited, processing begins. The effect of varying a single parameter, or small set of parameters, may be investigated using this mode.

Rerun mode is entered by typing the command

tvsim *pic* -r [*pars*]

where *pic* is the name of an input image datfile and *pars* is an optional ordinary file containing a predefined set of TVSIM parameters. Processing is immediately carried out using the parameters in the parameter file *pars* (if specified), or in the descriptor portion of *pic*. This mode is useful for regenerating a processed image without having to respecify the parameters.

5.1.2. TVSIM Components

The standard TVSIM processing programs are called *camera*, *prefilt*, *channel*, *postfilt*, and *display*. Each is described in more detail below.

5.1.2.1. The Camera

The video camera is characterized by the following list of parameters. Default values are listed in parentheses. Parameter values indicated by * are computed or set by the program.

c_proc: camera processing flag (*)
c_tlines: total number of lines per frame (V_{INPUT}/4)
c_alines: total number of active lines per frame (*)
c_frames: number of frames/sec (1)
c_ir: interlace ratio (1)
c_aper: aperture function (Gaussian)
c_amp: aperture amplitude (100)
c_vaper: aperture V size in mosaic elements (5)
c_haper: aperture H size in mosaic elements (5)

c_vpitch: aperture V pitch in mosaic elements (4)

c_hpitch: aperture H pitch in mosaic elements (1)

c_vsig: Gaussian V sigma in mosaic elements (1)

c_hsig: Gaussian H sigma in mosaic elements (1)

c_lens: lens point-spread function (ideal)

c_ar: camera aspect ratio (1)

c_hblank: percentage of line time for H blanking (0)

c_vblank: percentage of frame time for V blanking (0)

c_gam1: photoconductive gamma (1)

c_gam2: external gamma correction (1)

c_l: lens upsampling factor (*)

c_m: lens downsampling factor (*)

c_erase: charge readout mode: passive=0, linear=1, exp=2 (0)

c_apname: name of optional datfile filter for camera aperture (null)

A subset of these camera parameters are used to calculate an approximate video bandwidth for Kell factors of 1 and 0.7. The following formula is used [116]:

$$f_{MAX} = \frac{K * c\_tlines^2 * c\_ar * c\_frames * (1 - c\_vblank)}{2 * (1 - c\_hblank)} , \qquad (5.1)$$

where K is the Kell factor [117].

Based on the values of the camera parameters, the image may require cropping and/or spatial scaling in order to map the image points onto the camera mosaic. Cropping to the desired aspect ratio is performed by the program *ar_crop*, and scaling is performed by the program *lens*. Photoelectric conversion and scanning is performed by the program *camera*.

5.1.2.1.1. Aspect Ratio Crop

If the aspect ratio of the input image (H_{INPUT}/V_{INPUT}) differs from the desired aspect ratio, *c_ar*, by more than 5%, cropping is performed. The cropping program, *ar_crop*, crops *pic* to the largest size with the correct aspect ratio, centering the output. The dimensions of the cropped picture are $H_{CROP} \times V_{CROP}$. The output image is called *pic.ar*.

5.1.2.1.2. The Lens

If a lens PSF other than "ideal" is specified, the cropped image is convolved with the desired lens PSF. However, lens processing may be required even if an ideal lens is specified. This will occur if the area of the cropped image differs from the area of the camera mosaic by more than 5%. Sampling rate conversion using bilinear interpolation is performed to scale the dimensions of the cropped image to those of the camera mosaic. The dimensions of the camera mosaic are, ideally, $H_{CAM} \times V_{CAM}$, where V_{CAM} = *c_alines***c_vpitch* and $H_{CAM} = V_{CAM}$**c_ar*. The total number of *active* scanning lines is *c_alines*, computed from *c_tlines* and *c_vblank*. Spatial scaling is performed by interpolating the image in both directions by an integral factor of *c_l*, and then subsampling by an integral factor of *c_m*. These factors are chosen as the smallest integers producing the required mosaic dimensions. The output of the lens program is called *pic.lens*, and represents the focal plane illuminance distribution.

5.1.2.1.3. The Scanning Process

At this point, the object plane luminance has been turned into the focal plane illuminance, which must now be converted to charge and read out. This processing is done by the camera program. Light-to-charge conversion is performed by a simple tone scale transformation whose gamma is defined by *c_gam*1. The transformed mosaic elements, now interpreted as positive charge values, are ready to be sampled.

Either tube-type or solid-state cameras can be simulated by the proper choice of aperture parameters. Several common aperture functions are offered. A Gaussian aperture function with a horizontal pitch of 1 mosaic element simulates a Gaussian beam moving "continuously" across a charge distribution in the horizontal direction. Usually, a vertical pitch size of 4 mosaic elements is sufficient to model beam intensity variation between scan lines. A box-shaped (prism) aperture function of size $H_A \times V_A$ with pitch spacings $H_C \times V_C$ simulates the solid-state array geometry of Fig. 2.12. Gaussian aperture functions are specified by the additional parameters *c_hsig* and *c_vsig*, the horizontal and vertical sigma values.

The amplitude of the aperture function affects charge readout operation only if linear or exponential erasure is selected. As explained in Chapter 2, linear erasure subtracts the beam aperture distribution from the charge distribution upon which it falls. The sum of the erased values becomes the readout signal at that point. Exponential erasure multiplies the beam distribution by the charge distribution before subtraction, so beam amplitudes near unity should be selected. In both cases, a larger beam amplitude will erase more charge at each location, increasing the gain and extending the frequency response of the output signal. If passive linear filtering is selected, a convolution operation is performed, with the result of each sum-of-products normalized by the sum of the aperture coefficients. This mode of operation is appropriate for solid-state sensor simulation.

The user may supply the name of a datfile for the aperture function. In this case, the coefficients and dimensions are read from the external datfile and are used as the camera aperture.

If the interlace parameter *c_ir* is 1, progressive scanning is performed. If *c_ir* = N, then N:1-interlaced scanning is performed. Interlaced scan can be used to investigate the effects of destructive readout and aperture correction. To properly decode the interlaced video signal, the interlace parameter at the display should be equal to that of the camera (*d_ir* = *c_ir*).

The readout signal undergoes a second tone scale operation to simulate external gamma correction. The parameter *c_gam2* characterizes this transformation.

The gamma-corrected output is the video signal, which is called *pic.c* and is kept as a 2-D datfile for display purposes. The dimensions of the video signal datfile are $H_{VIDEO} \times V_{VIDEO} = H_{CAM}/c\_hpitch \times V_{CAM}/c\_vpitch$. At this point, the vertical size should be equal to *c_alines*, the number of active video lines specified. In relation to the dimensions of the camera mosaic, the video signal for a solid-state camera will be subsampled in *both* dimensions, whereas for a tube-type camera, it will be subsampled only *vertically*. If desired, the video signal can be stretched in either or both dimensions by the prefilter, which is described next.

5.1.2.2. The Prefilter

The prefilter can be used to spatially scale the video signal in one or both dimensions. Thus, it can be used as a standards converter, as a 2-D aperture corrector [118], or as a sample-and-hold filter for a solid-state camera [77]. The prefilter parameters are:

pr_proc: prefilter processing flag (*)
pr_filt: prefilter type (boxcar)
pr_vsize: vertical filter size (2)
pr_hsize: horizontal filter size (2)
pr_vl: vertical upsampling factor (1)
pr_vm: vertical downsampling factor (2)
pr_hl: horizontal upsampling factor (1)
pr_hm: horizontal downsampling factor (2)
pr_amp: filter amplitude (100)
pr_name: name of optional datfile prefilter (null)

The default values simulate a scan converter that halves the number of video lines sent into the channel. Aperture correction can be performed by choosing a sharpening mask for the filter type. A custom datfile filter can also be specified. The output signal is called *pic.pr* with dimensions $H_{PR} \times V_{PR} = H_{VIDEO} * pr\_hl/pr\_hm \times V_{VIDEO} * pr\_vl/pr\_vm$.

5.1.2.3. The Channel

A noisy analog channel is simulated; it may be used to lowpass filter the video signal and to simulate ghosting. The channel parameters are:

ch_proc: channel processing flag (*)
ch_filt: channel lowpass filter type (impulse)
ch_cutoff: channel normalized cutoff frequency (1)
ch_ghdel: ghosting delay in pels (0)
ch_ghamp: ghosting amplitude (0)
ch_snr: channel SNR in dB with additive white Gaussian noise (-1 = none)

The filter program convolves the channel impulse response with the 1-D input waveform. The channel impulse response is the convolution of the lowpass impulse response and the ghosting impulse response. White Gaussian noise is added to the filtered waveform; the resulting SNR is *ch_snr* dB. Typical zero-phase lowpass impulse responses are offered. A single multipath reflection (ghost) is offered; its delay and amplitude are specified by *ch_ghdel* and *ch_ghamp*. The output signal is called *pic.ch* and has the same dimensions as the input signal: $H_{CH} \times V_{CH} = H_{PR} \times V_{PR}$.

5.1.2.4. The Postfilter

The postfilter can be used to spatially scale the channel signal in one or both dimensions. Thus, it can be used as a standards converter, as a 2-D channel inverse filter, or as an inverse filter for the display aperture. The postfilter parameters are:

po_proc: postfilter processing flag (*)
po_filt: postfilter type (triangle)
po_vsize: vertical filter size (3)
po_hsize: horizontal filter size (3)
po_vl: vertical upsampling factor (2)

po_vm: vertical downsampling factor (1)

po_hl: horizontal upsampling factor (2)

po_hm: horizontal downsampling factor (1)

po_amp: filter amplitude (100)

po_name: name of optional datfile for postfilter (null)

The default values simulate a scan converter that uses linear interpolation to double the number of video lines received from the channel. A custom datfile filter may be used instead. The output signal is called *pic.po* with dimensions $H_{PO} \times V_{PO} = H_{VIDEO} * po\_hl / po\_hm \times V_{VIDEO} * po\_vl / po\_vm$.

5.1.2.5. The Display

The display program interpolates its input signal onto the display mosaic. The video display is characterized by the following list of parameters:

d_proc: display processing flag (*)

d_tlines: total number of lines per frame ($V_{INPUT}/4$)

d_alines: total number of active lines per frame (*)

d_frames: number of frames per second (1)

d_ir: interlace ratio (1)

d_aper: aperture function (Gaussian)

d_amp: aperture amplitude (100)

d_vaper: aperture V size in mosaic elements (9)

d_haper: aperture H size in mosaic elements (9)

d_vpitch: aperture V pitch in mosaic elements (4)

d_hpitch: aperture H pitch in mosaic elements (1)

d_vsig: Gaussian V sigma in mosaic elements (2)

d_hsig: Gaussian H sigma in mosaic elements (2)

d_brit: brightness offset (0)

d_cont: contrast scaling factor (*d_vpitch* * *d_hpitch*)

d_hblank: percentage of line time for H blanking (0)

d_vblank: percentage of frame time for V blanking (0)

d_ar: display aspect ratio (1)

d_gam: display gamma (1)

d_apname: name of optional datfile filter for display aperture (null)

As in the camera, the display program can simulate tube-type or solid-state devices by the proper choice of aperture parameters. For instance, a Gaussian aperture moving with a horizontal pitch of one element simulates tube-type scanning; a rectangular uniform aperture (prism) displaced by appropriate amounts horizontally and vertically simulates solid-state devices. A custom aperture function can also be specified. For normal operation, the display aspect ratio (*d_ar*) and interlace ratio (*d_ir*) should equal those of the camera. The input signal first undergoes a tone scale transformation characterized by *d_gam*. Then the input signal is zero-padded horizontally by *d_hpitch* and vertically by *d_vpitch*. The zero-padded signal is then linearly filtered by the display aperture function. Each convolution sum is divided by the sum of the filter coefficients and multiplied by a contrast scaling factor (*d_cont*). The default contrast scaling factor is the product of the zero-padding factors; more or less contrast can be achieved by specifying a number smaller or larger than the default. A brightness offset (*d_brit*) can be added to the resulting luminance value to brighten or darken the image.

If a Gaussian aperture is specified, the sigma and pitch values are used to calculate the blanked-line and flat-field contrast and modulation depth [68]. These concepts are explained more fully in Appendix C and modulation depth curves are shown in Fig. 5.3.

5.2. TVSIM Experiments

This chapter concludes with some experiments designed to show the power and flexibility of the TV simulation program.

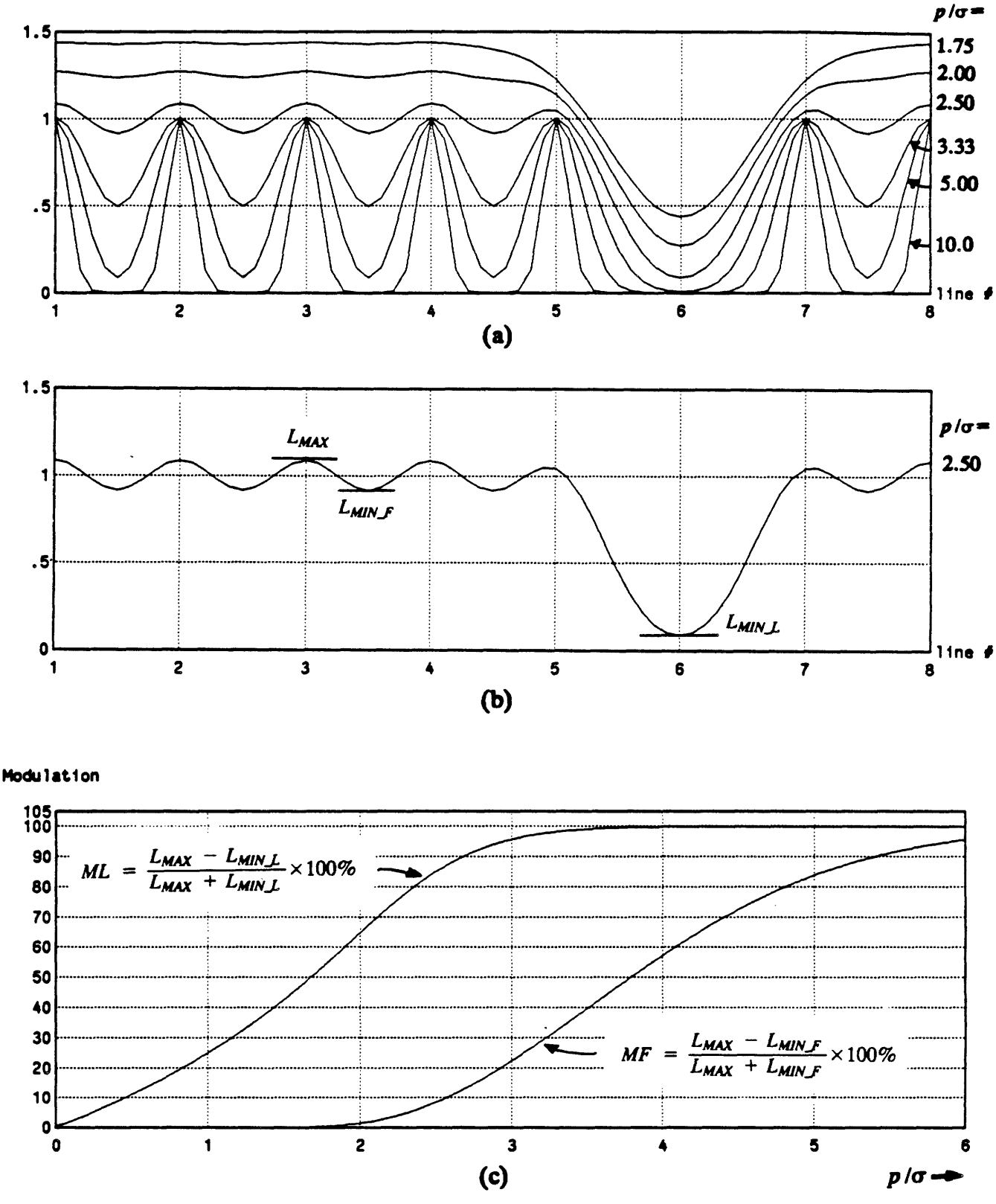


Fig. 5.3. (a) Effect of p/σ ratio on modulation of flat field and single blanked line. (b) Luminance values used to determine modulation depth. (c) Modulation depth curves for flat field (*MF*) and single blanked line (*ML*).

5.2.1. 64-line TV Simulations

The advent of solid-state cameras and displays is causing a renewed interest in image quality because the reproduced pictures have a different appearance. For instance, aliasing may occur in both spatial dimensions when CCD cameras are used, and pixel structure is evident in solid-state displays. A modern video system can assume any combination of camera and display: (1) tube-type to tube-type (TT); (2) tube-type to solid-state (TS); (3) solid-state to tube-type (ST); and (4) solid-state to solid-state (SS).

Fig. 5.4 shows 64-line simulations of each of these four combinations. The tube-type camera is linear with a Gaussian read beam ($\sigma_r = 0.25$ TVL). The tube-type display is linear with a Gaussian write beam ($\sigma_w = 0.375$ TVL). The solid-state camera is linear with 3×3 active area within a 4×4 resolution cell. The solid-state display is linear with 4×4 active area within a 4×4 resolution cell. Fig. 5.4(a) is the TT simulation, showing vertical aliasing only, as expected. Fig. 5.4(b) is the ST simulation, showing aliasing in both directions because of the discrete 2-D sampling of the CCD camera. Fig. 5.4(c) is the TS simulation, showing vertical aliasing and a pronounced sampling structure. Fig. 5.4(d) is the SS simulation, showing aliasing in both directions, as well as a visible sampling structure.

On close inspection, Fig. 5.4(b) seems the most pleasing, since the aliasing artifacts are symmetrical, and the scan lines impart a less obtrusive sampling structure than the square display pixels. When viewed at a distance such that two successive scan lines subtend an angle of 2 minutes of arc (about 8 ft. away from the page), the sampling structure becomes nearly invisible. However, aliasing is still visible, and in the two pictures on the right, the aliasing artifacts are symmetrical in both directions.

5.2.2. High-Resolution Simulations

Unlike other video simulations described in the literature, TVSIM can model systems of arbitrary resolution. A large amount of virtual memory is the only requirement for high-resolution simulations. An example will make this clear.

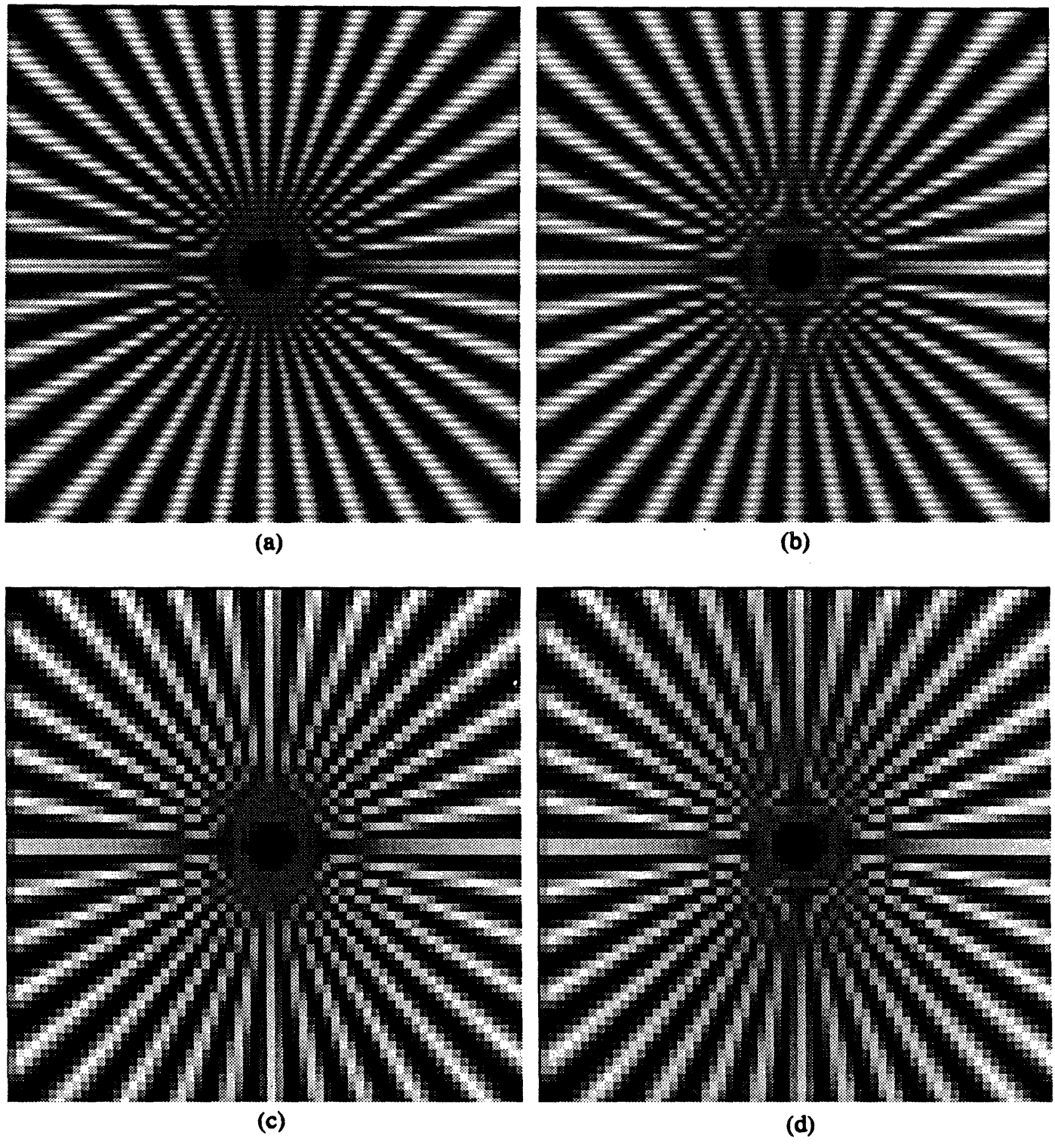


Fig. 5.4. Simulation of four combinations of cameras and displays. (a) Tube-type to tube-type. (b) Solid-state to tube-type. (c) Tube-type to solid-state. (d) Solid-state to solid-state.

Suppose one desires an accurate full-frame NTSC simulation. Since there are 483 active scan lines, four times this number, or 1932, is a good choice for the vertical dimension of the camera and display mosaic. This sampling ratio yields adequate aperture discretization and allows a fine-grained scan line structure. The horizontal dimension, assuming a 4:3 aspect ratio, is 2576 mosaic elements. Aperture sizes may be as large as 15 elements vertically, which means that 15 lines of charge must fit in main memory. At 4 bytes/element, this yields a memory size of 155 Kbytes, not counting overhead. Most fixed memory computers cannot handle this volume of data. The following high-resolution simulations were performed on a virtual memory computer (VAX 11/785).

5.2.2.1. Relative Quality of NTSC vs. HDTV

TVSIM can be used to compare the quality of systems with different scanning standards. Of special interest is the relative quality of NTSC and HDTV systems. In this and subsequent sections, a HDTV system is assumed to have *twice* the number of scan lines per frame as the NTSC system. Starting with a high-resolution synthetic charge image (2576H × 1932V), the NTSC and HDTV cameras linearly filter with a Gaussian read beam ($\sigma_r = 0.25$ TVL) and subsample vertically by factors of four and two, respectively, yielding 483 and 966 active video lines. The linear NTSC and HDTV CRT's vertically interpolate by factors of four and two and filter the video signal with a Gaussian write beam ($\sigma_w = 0.375$ TVL). The simulated output pictures are the same size as the original charge image, and to display them at full resolution using a 100 line-per-inch halftone screen would require an image size of 36×27 inches. Instead, the central portion of the displayed image is cropped and shown at full resolution in Fig. 5.5. Approximately 110 lines of NTSC video are shown in the left half of the image; 220 lines of HDTV are shown on the right. It is readily apparent that HDTV displays a better picture. Vertical aliasing is hardly noticeable, and scan lines are thinner and less visible. An uncompressed HDTV signal, however, requires at least four times the video bandwidth of NTSC. Enhancing the picture quality of standard NTSC is an alternative to HDTV, and is the subject of the next section.

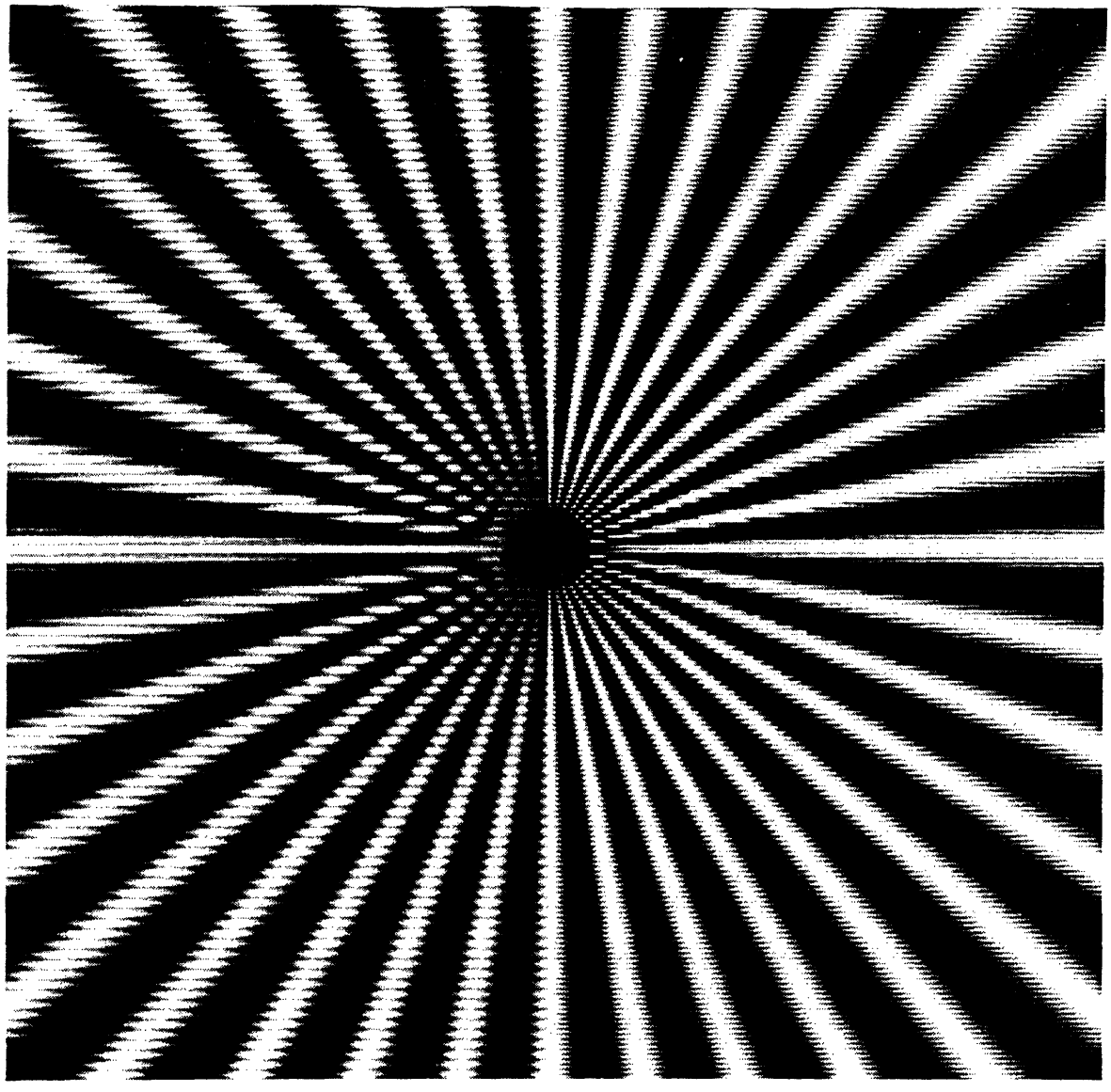


Fig. 5.5. Effect of number of scan lines on image quality. (Left half) NTSC quality. (Right half) HDTV quality.

5.2.2.2. EDTV Simulations

Fig. 5.5 showed that reducing the scan line visibility results in improved picture quality. Line visibility can be reduced in a number of ways. Chapter 4 described adaptive beam reconstruction and adaptive line interpolation. Simpler nonadaptive methods are discussed here.

5.2.2.2.1. Line Interpolation with Sharpening

Line interpolation can be used to synthesize new lines in the displayed picture. Each scan line is thinner and less visible, so picture quality is improved. Since digital hardware must be used to perform line interpolation, a small amount of 2-D sharpening can be performed to further improve the displayed picture.

The effect of nonadaptive 2:1 interpolation using line averaging is shown in Fig. 5.6. In (a) is shown the original charge image. NTSC quality is simulated in (b), and 2:1 line-interpolated NTSC without sharpening is shown in (c). HDTV quality is shown in (d). Although aliasing is still evident in (c), the line visibility is less. The resulting picture quality is between that of NTSC and HDTV.

Aliasing is rarely objectionable in natural imagery, so simple line interpolation can only improve picture quality. Adaptive line interpolation [111] can produce better results when motion is present. Sharpening the video signal after line interpolation can further improve the quality. An example is shown in Fig. 5.7. NTSC quality is simulated in (a). Sharpening the 2:1 line-interpolated NTSC signal results in (b). HDTV quality is simulated in (c). Similar enhancement of the HDTV signal results in (d). These pictures show that simple 2:1 line interpolation with sharpening results in improved picture quality.

5.2.2.2.2. Wobble Raster

Improved image quality can be achieved by dithering the electron beam as it moves across the screen. Wobble raster has been used to increase the resolution of standard displays [119, 120]. However, it can also be used to reduce line visibility

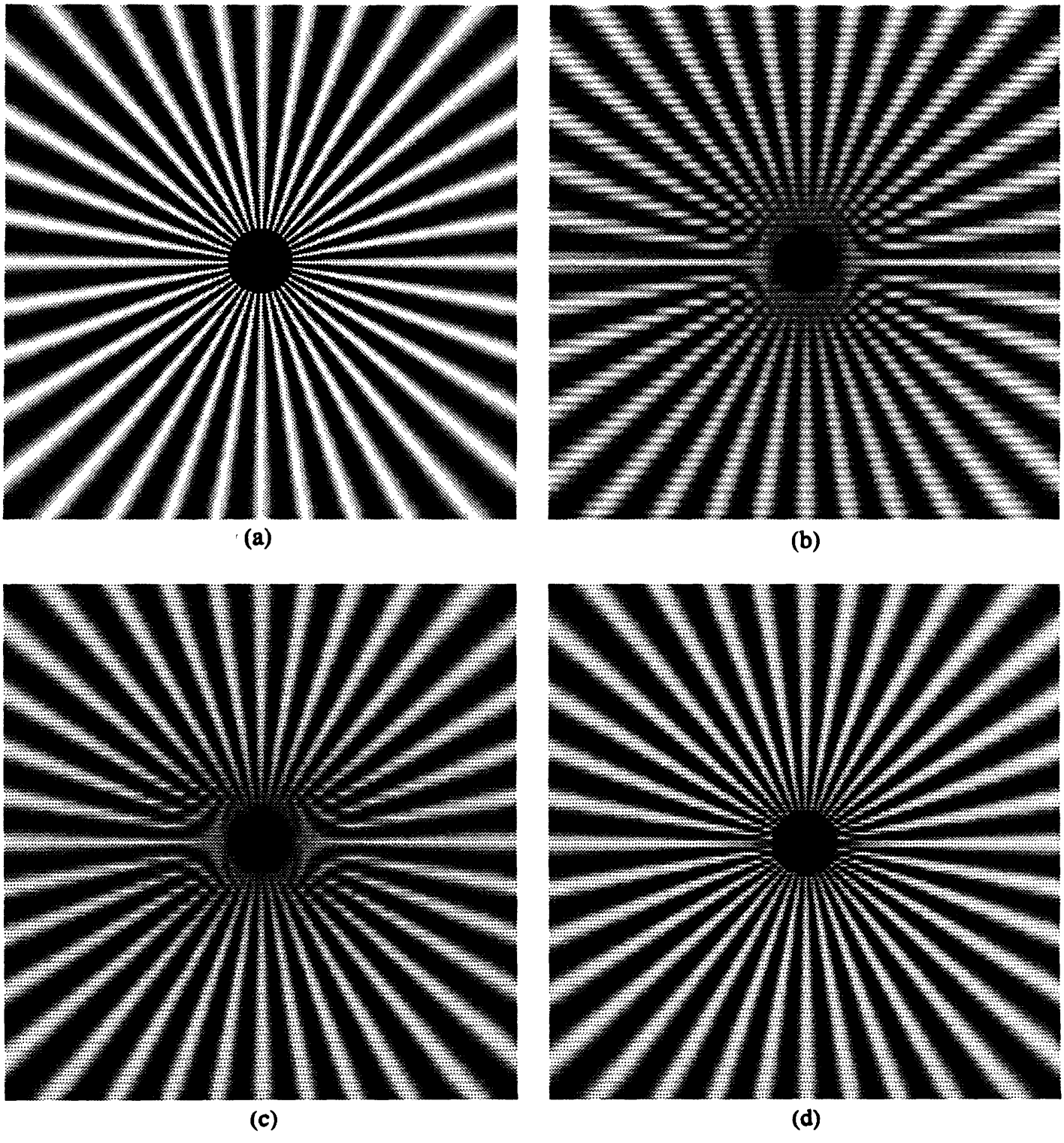


Fig. 5.6. Effect of line interpolation on image quality. (a) Original. (b) NTSC quality. (c) 2:1 line-interpolated NTSC. (d) HDTV quality.

Fig. 5.7. Effect of line interpolation and sharpening on image quality. (a) NTSC quality. (b) 2:1 line-interpolated NTSC with sharpening. (c) HDTV quality. (d) 2:1 line-interpolated HDTV with sharpening.

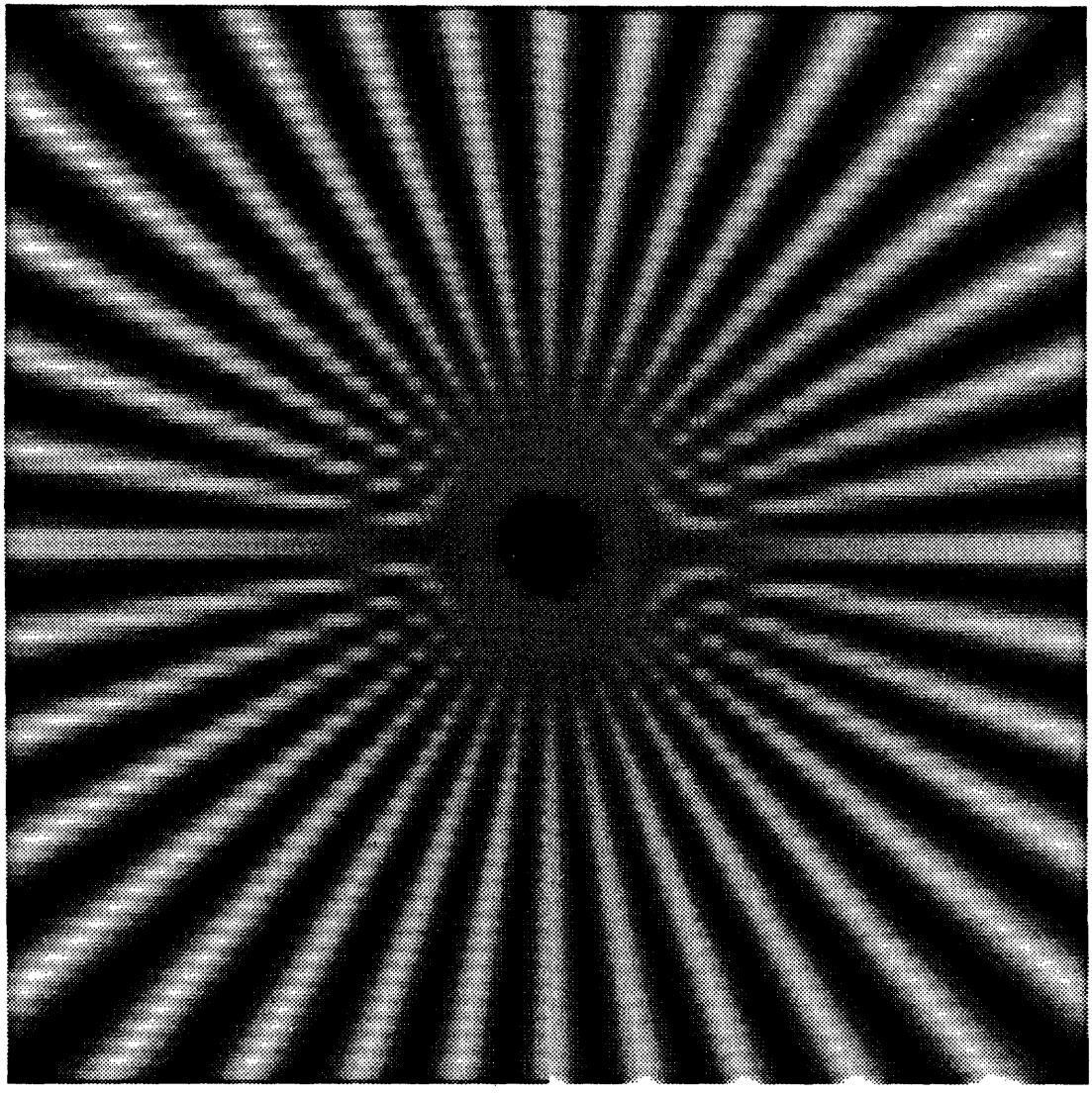


Fig. 5.8. Improved image quality using wobble raster. (Left half) wobble off. (Right half) wobble on.

by converting the scan line structure into a crosshatch pattern that is less visible to the eye. The frequency, phase, and amplitude of the wobble are adjusted so that the maximum slope is $\pm$ 45^c, with identical phase from line to line. This is simulated in Fig. 5.8. The left side of the image is displayed with no wobble, the right side with wobble. Line visibility is decreased, yet there is no detectable distortion or loss of resolution.

5.3. Summary

This chapter described the operation of TVSIM, a program that simulates the important spatial characteristics of a monochrome baseband TV system. Several experiments showed the power and flexibility of the program. TVSIM was used to simulate the quality NTSC, EDTV, and HDTV systems. Line interpolation with sharpening results in an enhanced NTSC picture that is close to HDTV in quality.

Chapter 6

Conclusions and Suggestions for Further Research

This thesis has modeled the television process. It has (1) applied functional modeling to the spatiotemporal processes in the television chain, (2) used temporal integration patterns to explain and simulate motion rendition, (3) incorporated HVS characteristics to predict subjective response, (4) proposed several new algorithms for the real-time reconstruction of video signals, and (5) compared standard-, enhanced-, and high-definition TV pictures. A detailed summary follows, with suggestions for future research.

Chapter 2 applied functional modeling to the spatial processes of a monochrome baseband television system. A general camera lens was modeled as a linear space-variant filter. Various charge scanning models were developed, and the beam sharpening effect was verified by numerical simulation on natural and synthetic imagery. Linear erasure has the highest sharpening, but the most distortion. It can also be used to simulate the steady-state appearance of images scanned by low beam currents. The effect of beam width on image quality was investigated and it was shown that for natural imagery, the read beam sigma should be approximately 0.2 TVL and the write beam sigma should be about 0.4 TVL for good subjective results. Aliasing and blur were studied in both the spatial and frequency domains. The effect of square uniform aperture sizes on image quality was studied. To reduce aliasing, the camera fill ratio should be greater than 50%; to reduce sampling structure visibility, the display fill ratio should be greater than 90%. Finally, a linear bandpass model for the HVS predicted greatly reduced scan line visibility and slightly reduced aliasing visibility at normal viewing distances.

Chapter 3 examined the temporal processes of a single image element from camera to display. It was shown how camera lag mechanisms arising from different physical processes can produce similar video signal waveforms. Two models of charge readout were proposed and simulated. The exponential model accurately simulates the exponential lag behavior of many tube-type cameras, and also yields stable charge dynamics. The exponential readout model was improved by

introducing a dependence of readout efficiency on available charge. Signal-dependent lag and bias light improvement was simulated using the extended model. The 1-D model also included a temporal model of the HVS, which predicted decreased flicker sensitivity at 60 and 90 Hz. A simple 3-D television model was developed by considering the temporal integration patterns of all elements as well as charge spreading between elements. The relationship between scanning rates, lag, and image retention was demonstrated, and the motion rendition of various imaging devices was simulated. It was shown that the shape of a moving object deforms when raster scanned, and the severity of deformation depends on the object's velocity.

Chapter 4 showed that adaptive filtering can improve picture quality by reducing sampling structure visibility and aliasing, while retaining important image detail. Analog beam shaping was first considered. This method exploits knowledge of edge orientation and strength. A simple algorithm was described that determines edge orientation with high precision, and works well even in the presence of noise. Three measures of edge strength were proposed. The simplest, slope averaging, provides a method for smoothly varying the filter eccentricity across an edge. It also behaves well in the presence of noise. Experiments with adaptive Gaussian interpolation of natural imagery suggest that edge-finding algorithms may work better if edges are located *after* bilinear interpolation of the received signal. A method of adaptive line interpolation using sharpened Gaussian filters was described. A new block-matching method was proposed for adaptive 2:1 line interpolation. This technique can produce a high-quality frame of video from each field. Thus, 60 progressively scanned frames per second can replace 30 interlaced frames. The parameters used in this simulation ($B=7$, $R=1$) require 21 subtractions, 19 additions, 1 division, and 2 comparisons per interpolated point. Exploiting parallelism, this computation is feasible at real-time rates and could be performed using a single IC chip. Extending the algorithm to subpixel accuracy can improve the performance.

Chapter 5 described the operation of TVSIM, a program that simulates the important spatial characteristics of a monochrome baseband TV system. Several experiments showed the power and flexibility of the program. TVSIM was used to

simulate the quality of NTSC, EDTV, and HDTV. Line interpolation with sharpening resulted in an enhanced NTSC picture that is close to HDTV in quality.

The scope of this thesis is quite large. Only a few of the most important parameters were considered in the modeling of each component. This chapter concludes by suggesting some extensions to this work.

First, the physical properties of TV components, especially the camera and the display, can be modeled in more detail. In the camera, the light-to-charge conversion can be extended to include trapping and recombination effects, beam-blocking contacts, photon fluctuations, shot noise, and dependence on temperature and target voltage. More accurate models of the charge readout process can be developed. In camera tubes, the readout process can be extended to model beam bending and to include variations in beam shape and in readout efficiency. In solid-state sensors, charge transfer inefficiencies and blooming can be incorporated. Highlight suppression can also be modeled. In the display, blooming, geometric distortion, and halo effects can be included.

Second, numerical models of the HVS should be developed and linked to TV simulation programs, such as TVSIM. This will aid in the understanding of the human visual system, and will provide a means to predict the subjective response to TV imagery.

Third, the basic TVSIM model can be extended by adding new video processing modules. For instance, AM or FM modulation programs can be used to study the effect of RF degradations on image quality. The effect of camera and display parameters on the performance of video compression or enhancement algorithms can be investigated. TVSIM should also be extended to operate on monochrome image sequences, so that temporal effects, such as camera lag, can be studied. Another important extension is to develop numerical models for color cameras, displays, and component coding.

Finally, TVSIM should be modified for adaptive processing in all components. Chapter 4 described adaptive interpolation techniques at the display. Adaptive sharpening techniques at the display should be modeled, and adaptive prefiltering at the camera should be considered, too. The effect of adaptive prefiltering on adaptive

postfiltering can then be studied. Adaptive processing holds much promise and is easily simulated numerically; it should be considered an essential part of future television systems.

References

[1] O.H. Schade, "On the Quality of Color Television Images and the Perception of Color Detail," *SMPTE J.*, vol. 67, no. 12, pp. 801-819, Dec. 1958.

[2] D. Teichner, "Quality Improvement by Adaptive Inter-/Intraframe Processing in PAL TV Receivers," *IEEE Trans. Cons. Elect.*, vol. 31, no. 3, pp. 226-239, Aug. 1985.

[3] A. Roberts, "The Improved Display of 625-Line Television Pictures: Adaptive Interpolation," BBC RD 1985/5, May 1985.

[4] V. Tom, "Adaptive Filter Techniques for Digital Image Enhancement," *SPIE*, vol. 588, 1985.

[5] J.L.E. Baldwin, "Enhanced Television - Progressive Experience," *SMPTE J.*, vol. 94, no. 9, pp. 904-913, Sept. 1985.

[6] Y. Ninomiya, Y. Ohtsuka, and Y. Izumi, "A Single Channel HDTV Broadcast System - The Muse," *NHK Lab. Note*, no. 304, pp. 1-12, Sep. 1984.

[7] R.D. Kell, A.V. Bedford, and G.L. Fredendall, "A Determination of Optimum Number of Lines in a Television System," *RCA Rev.*, vol. 5, no. 1, pp. 8-30, July 1940.

[8] G.E. Anner, *Elements of Television Systems,* New York: Prentice-Hall, 1951.

[9] D.G. Fink, *Principles of Television Engineering,* New York: McGraw-Hill, 1940.

[10] L.A. Selke, "Mathematical Model to Describe Vidicon Operation," *IEEE Trans. Electr. Dev.*, vol. 16, no. 7, pp. 625-631, July 1969.

[11] O.H. Schade, "Electron Optics and Signal Read-Out of High-Definition Return Beam Vidicon Cameras," *RCA Rev.*, pp. 60-119, Mar. 1970.

[12] A.A. Guida, K. Jonnalagadda, D. Raychaudhuri, and L. Schiff, "Computer Modeling in the RCA Satcom System," *RCA Engineer*, vol. 27, no. 6, pp. 84-91, Nov./Dec. 1982.

[13] R.J. Klensch and H. Waldman, "New Simulation Techniques for Video Provide Powerful Investigative Tools," *RCA Engineer*, vol. 30, no. 1, pp. 38-47, Jan./Feb. 1985.

[14] P. Mertz and F. Gray, "A Theory of Scanning and its Relation to the Characteristics of the Transmitted Signal in Telephotography and Television," *Bell Syst. Tech. J.*, vol. 13, pp. 464-515, July 1934.

[15] F.O. Huck, N. Halyo, and S.K. Park, "Aliasing and Blurring in 2-D Sampled Imagery," *Applied Optics*, vol. 19, no. 13, pp. 2174-2181, July 1980.

[16] M. Potmesil and I. Chakravarty, "A Lens and Aperture Camera Model for Synthetic Image Generation," *Comput. Graphics*, vol. 15, no. 3, Aug. 1981.

[17] A. Miller and J.R. Izatt, "Destructive Readout in Image Tubes," *Applied Optics*, vol. 5, no. 12, pp. 1940-1945, Dec. 1966.

[18] M. Kurashige, "Effect of Self-Sharpening in Low-Velocity Electron-Beam Scanning," *IEEE Trans. Electr. Dev.*, vol. 29, no. 10, pp. 1570-1579, Oct. 1982.

[19] O.H. Schade, "The Resolving Power Functions and Quantum Processes of Television Cameras," *RCA Rev.*, pp. 460-535, Sept. 1967.

[20] O.H. Schade, "Resolving Power Functions and Integrals of High-Definition Television and Photographic Cameras - A New Concept of Image Evaluation," *RCA Rev.*, vol. 32, pp. 567-609, Dec. 1971.

[21] L.J. VanDePolder, "Target-Stabilization Effects in Television Pick-Up Tubes," *Philips Res. Repts.*, vol. 22, pp. 178-207, 1967.

[22] S. Nudelman, "The Detectivity of Electron Beam Scanning Types of Image Tubes," *Applied Optics*, vol. 6, no. 1, pp. 149-157, Jan. 1967.

[23] C.F. Wolcott, "Problems in Television Image Resolution," *SMPTE J.*, vol. 36, pp. 65-81, Jan. 1941.

[24] E.W. Engstrom, "A Study of Television Image Characteristics, Part 1:," *Proc. IRE*, vol. 21, no. 12, pp. 1631-1651, Dec. 1933.

[25] A.P. Pica, "Simulation of Alphanumeric Characters on Color Monitors," *SID Digest*, pp. 58-59, 1984.

[26] O.H. Schade, "Image Reproduction by a Line Raster Process" in *Perception of Displayed Information*, ed. L.M. Biberman, Plenum Press, 1973, pp. 233-278.

[27] E.H. Adelson, C.R. Carlson, and A.P. Pica, "Modeling the Human Visual System," *RCA Engineer*, vol. 27, no. 6, pp. 56-64, Nov./Dec. 1982.

[28] C.R. Carlson and R.W. Cohen, "A Simple Psychophysical Model for Predicting the Visibility of Displayed Information," *Proc. SID*, vol. 21, no. 3, pp. 229-245, 1980.

[29] R.W. Klopfenstein, A.P. Pica, and C.R. Carlson, "Perceptual Analysis and Simulation of Color Monitor Displays," *RCA Engineer*, vol. 29, no. 6, pp. 28-32, Nov./Dec. 1985.

[30] S.S. Perlman, "Computer Simulation of Horizontal Transient Response of the NTSC Color-TV System," *RCA Rev.*, vol. 42, pp. 463-477, Sept. 1981.

[31] L.C. Palmer and A. Shenoy, "Simulation of TV Transmission Over the Communications Satellite Channel," *IEEE Trans. Commun.*, vol. 34, no. 2, pp. 188-199, Feb. 1986.

[32] K.M. Wong, "Transient Oscillations and Noise in the Reception of TV Signals," *IEEE Trans. Broadcast.*, vol. 30, no. 2, pp. 29-37, June 1984.

[33] A.M. Dhake, *Television Engineering*, New York: McGraw-Hill, 1979.

[34] S. Kato, Y. Nonaka, M. Maruyama, and C. Oguso, "Performance Characteristics of Improved Pickup Tube: New Diode Gun Saticon," *SMPTE J.*, vol. 92, no. 10, pp. 1041-1046, Oct. 1983.

[35] E.F. DeHaan, A.V.D. Drift, and P.P.M. Schampers, "The Plumbicon, A New Television Camera Tube," *Philips Tech. Rev.*, vol. 25, no. 6/7, pp. 133-151, July 1964.

[36] S.A. Ochs and P.K. Weimer, "Some New Structure-Type Targets for the Vidicon - An Analysis of Their Operation," *RCA Rev.*, pp. 48-61, Mar. 1958.

[37] B.P. Miller, G.A. Beck, and J.M. Barletta, "Performance Evaluation of the Two-Inch Return-Beam Vidicon Three-Camera Subsystem," *SMPTE J.*, vol. 81, pp. 105-111, Feb. 1972.

[38] A.D. Cope, S. Gray, and E.C. Hutter, "The Television Camera Tube as a System Component" in *Photoelectronic Imaging Devices, Vol. 2*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 15-51.

[39] R.W. Redington, "Introduction to the Vidicon Family of Tubes" in *Photoelectric Imaging Devices: Vol. 2*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 263-273.

[40] D.H. Ballard and C.M. Brown, *Computer Vision,* Prentice-Hall, 1982.

[41] K.R. Castleman, *Digital Image Processing,* Prentice-Hall, 1979.

[42] J.W. Goodman, *Introduction to Fourier Optics,* McGraw-Hill, 1968.

[43] M. Potmesil and I. Chakravarty, "A Lens and Aperture Camera Model for Synthetic Image Generation," *Comput. Graphics*, vol. 15, no. 3, Aug. 1981.

[44] E. Hecht and A. Zajac, *Optics,* Addison-Wesley, 1979.

[45] W.W. Frame, "Minimum Resolvable and Minimum Detectable Contrast Prediction for Vidicon Cameras," *SMPTE J.*, pp. 21-26, Jan. 1985.

[46] R.E. Crochiere and L.R. Rabiner, *Multirate Digital Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall, 1983.

[47] J.N. Ratzel, "The Discrete Representation of Spatially Continuous Images," Ph.D. Thesis, MIT Dept. of Electr. Eng., Aug. 1980.

[48] D.G. Fink, *Principles of Television Engineering,* New York: McGraw-Hill, 1940.

[49] R.E. Flory, "Image Acquisition Technology," *Proc. IEEE*, vol. 73, no. 4, pp. 613-637, Apr. 1985.

[50] V.K. Zworykin, "The Iconoscope - A Modern Version of the Electric Eye," *Proc. IRE*, vol. 22, no. 1, pp. 16-32, Jan. 1934.

[51] H. Kiess, "The Physics of Electrical Charging and Discharging of Semiconductors," *RCA Rev.*, vol. 36, pp. 667-710, Dec. 1975.

[52] B. Meltzer and P.L. Holmes, "Beam Temperature, Discharge Lag and Target Biasing in Some Television Pick-Up Tubes," *British J. of Applied Physics*, vol. 9, pp. 139-143, Apr. 1958.

[53] S. Nudelman, "The Detectivity of Electron Beam Scanning Types of Image Tubes," *Applied Optics*, vol. 6, no. 1, pp. 149-157, Jan. 1967.

[54] O.H. Schade, "Electron Optics and Signal Read-Out of High-Definition Return Beam Vidicon Cameras," *RCA Rev.*, pp. 60-119, Mar. 1970.

[55] E.H. Stupp and R.S. Levitt, "The Plumbicon" in *Photoelectric Imaging Devices: Vol. 2*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 275-300.

[56] D.E. Pearson, *Transmission and Display of Pictorial Information,* New York: Wiley, 1975.

[57] G. Sadasiv, "Photoconductivity" in *Photoelectric Imaging Devices, Vol. I*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 111-146.

[58] W.F. Schreiber, "The Effect of Scanning Speed on the Signal/Noise Ratio of Camera Tubes," *Proc. IEEE*, vol. 52, no. 2, Feb. 1964.

[59] R.G. Neuhauser, "The Saticon Color Television Camera Tube," *SMPTE J.*, vol. 87, no. 3, pp. 147-152, Mar. 1978.

[60] A. Miller and J.R. Izatt, "Destructive Readout in Image Tubes," *Applied Optics*, vol. 5, no. 12, pp. 1940-1945, Dec. 1966.

[61] T.G. Schut, "Resolution Measurements on Camera Tubes," *SMPTE J.*, vol. 92, no. 12, pp. 1287-1293, Dec. 1983.

[62] S.J. Peppiatt and H.O.W. Jordan, "A 30mm High-Resolution Diode Gun Leddicon," *SMPTE J.*, vol. 93, no. 10, pp. 911-914, Oct. 1984.

[63] M. Kurashige, "Effect of Self-Sharpening in Low-Velocity Electron-Beam Scanning," *IEEE Trans. Electr. Dev.*, vol. 29, no. 10, pp. 1570-1579, Oct. 1982.

[64] E.H. Adelson and J.R. Bergen, "Spatiotemporal Energy Models for the Perception of Motion," *J. Opt. Soc. Am.*, vol. 2, no. 2, pp. 284-299, Feb. 1985.

[65] C.F. Hall and E.L. Hall, "A Nonlinear Model for the Spatial Characteristics of the Human Visual System," *IEEE Trans. Syst., Man, Cyber.*, vol. 7, no. 3, pp. 161-170, Mar. 1977.

[66] G.E. Anner, "The Optimum Number of Lines - Chapter 10" in *Elements of Television Systems*, ed. G.E. Anner, Prentice-Hall, 1951, pp. 374-394.

[67] C. Infante, "On the Resolution of Raster-Scanned CRT Displays," *Proc. SID*, vol. 26, no. 1, pp. 23-36, 1985.

[68] L.J.W. Versnel, "High-Resolution Monitor Tube," *Electronic Components and Applications*, vol. 5, no. 2, pp. 108-112, Feb. 1985.

[69] L.J. Pinson, "A Quantitative Measure of the Effects of Aliasing on Raster Scanned Imagery," *IEEE Trans. Syst., Man, Cybernetics*, vol. 8, no. 10, pp. 774-778, Oct. 1978.

[70] R. Legault, "Visual Detection Process for Electrooptical Images: Man - The Final Stage of an Electrooptical System" in *Photoelectric Imaging Devices, Vol.*

I, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 69-87.

[71] W. Lawson, "Electrooptical System Evaluation" in *Photoelectric Imaging Devices, Vol. I*, ed. L.M. Biberman, Plenum Press, 1971, pp. 375-409.

[72] A. Rose, "Vision: Human and Electronic," *Electron. Imaging*, pp. 56-61, Aug. 1983.

[73] J.A. Hall, "Evaluation of Signal-Generating Image Tubes" in *Photoelectric Imaging Devices, Vol. 2*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 77-115.

[74] N. Goto, Y. Isozaki, K. Shidira, E. Maruyama, T. Hirai, and T. Fujita, "Saticon: A New Photoconductive Camera Tube with Se-As-Te Target," *IEEE Trans. Electron Devices*, vol. 21, no. 11, pp. 662-666, Nov. 1974.

[75] R.G. Neuhauser, "Lag Reduction and Lag Characteristics of Television Camera Tube Signals" in *Television Technology in the 80's*, ed. J.F. Christensen, SMPTE, 1981, pp. 132-141.

[76] K.B. Benson, "A Brief History of Television Camera Tubes," *SMPTE J.*, pp. 708-712, Aug. 1981.

[77] H.E. Murphy, "Performance Characteristics of a Producible NTSC-Compatible Charge-Coupled Device (CCD) Image Sensor," *SPIE J.*, vol. 203, pp. 80-87, 1979.

[78] E. Dubois, "The Sampling and Reconstruction of Time-Varying Imagery with Application to Video Systems," *Proc. IEEE*, vol. 73, no. 4, pp. 502-522, Apr. 1985.

[79] T.N. Cornsweet, *Visual Perception*, New York: Academic Press, 1970.

[80] D.H. Kelly and R.E. Savoie, "Theory of Flicker and Transient Responses. III. An Essential Nonlinearity," *J. Opt. Soc. Am.*, vol. 68, no. 11, pp. 1481-1490, Nov. 1978.

[81] W.K. Pratt, *Digital Image Processing*, John Wiley & Sons, 1978.

[82] A. Franken, "A New High Resolution Plumbicon Tube" in *Television Technology in the 80's*, ed. J.F. Christensen, SMPTE, 1981, pp. 124-131.

[83] T. Ninomiya, K. Wakui, N. Goto, and K. Shidara, "Saticon and its Application to Small-Sized Color TV Cameras," NHK Tech. Monograph 25, pp. 1-39, Mar. 1976.

[84] T.M. Gurley and C.J. Haslett, "Resolution Considerations in Using CCD Imagers in Broadcast-Quality Cameras," *SMPTE J.*, vol. 94, no. 9, pp. 882-895, Sept. 1985.

[85] M. Collet, "Make Way for Solid-State Image Sensors," *Photonics Spectra*, pp. 103-113, Sept. 1985.

[86] R.G. Neuhauser, "Sensitivity and Motion Capturing Ability of Television Camera Tubes," *SMPTE J.*, vol. 68, no. 7, pp. 455-461, July 1959.

[87] G.W. Hughes, "Electronic Imaging With CCD's," *RCA Engineer*, vol. 29, no. 6, pp. 4-10, Nov./Dec. 1984.

[88] J.M. Younse, J.F. Breitzmann, and J.W. Freeman, "The Modular Charge-Coupled Device (CCD) Camera," *SPIE J.*, vol. 203, pp. 88-95, 1979.

[89] A.M. Goodman, "An Approximate Model of the Beam-Blocking Contact in a PbO Vidicon," *RCA Rev.*, vol. 36, pp. 408-424, Sept. 1975.

[90] M.H. Crowell and E.F. Labuda, "The Silicon-Diode-Array Camera Tube" in *Photoelectric Imaging Devices: Vol. 2*, ed. L.M. Biberman and S. Nudelman, Plenum Press, 1971, pp. 301-343.

[91] G.F. Simmons, *Differential Equations with Applications and Historical Notes*, McGraw-Hill, 1972.

[92] C.E. Froberg, *Numerical Mathematics: Theory and Computer Applications*, Benjamin-Cummings, 1985.

[93] F.O. Huck, N. Halyo, and S.K. Park, "Aliasing and Blurring in 2-D Sampled Imagery," *Applied Optics*, vol. 19, no. 13, pp. 2174-2181, July 1980.

[94] J.L.E. Baldwin, "Enhanced Television - Progressive Experience," *SMPTE J.*, vol. 94, no. 9, pp. 904-913, Sept. 1985.

[95] T. Tohyama, "A New 5.3-inch High-Brightness Beam-Index Display," *IEEE Trans. Cons. Elect.*, vol. 31, no. 3, pp. 174-183, Aug. 1985.

[96] M. Achiha, K. Ishikura, and T. Fukinuki, "A Motion-Adaptive High-Definition Converter for NTSC Color TV Signals," *SMPTE J.*, vol. 93, no. 5, pp. 470-476, May 1984.

[97] D.E. Dudgeon and R.M. Mersereau, *Multidimensional Digital Signal Processing*, Prentice-Hall, 1984.

[98] C.U. Lee, "Image Rotation by One-Dimensional Filtering," S.M. Thesis, MIT Dept. of Electr. Eng., Jan. 1985.

[99] A.J.E.M. Janssen, R.N.J. Veldhuis, and L.B. Vries, "Adaptive Interpolation of Discrete-Time Signals That Can Be Modeled as Autoregressive Processes," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. 34, no. 2, pp. 317-330, Apr. 1986.

[100] V. Tom, "Adaptive Filter Techniques for Digital Image Enhancement," *SPIE*, vol. 588, 1985.

[101] R. Wilson, H.E. Knutsson, and G.H. Granlund, "Anisotropic Nonstationary Image Estimation and Its Applications: Part II - Predictive Image Coding," *IEEE Trans. Commun.*, vol. 31, no. 3, pp. 398-406, Mar. 1983.

[102] A. Ikonomopoulos and M. Kunt, "High Compression Image Coding Via Directional Filtering," *Signal Processing*, vol. 8, pp. 179-203, 1985.

[103] M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second-Generation Image-Coding Techniques," *Proc. IEEE*, vol. 73, no. 4, pp. 549-574, Apr. 1985.

[104]G.B. Thomas, *Calculus and Analytic Geometry,* Addison-Wesley, 1972.

[105]R. Machuca and A.L. Gilbert, "Finding Edges in Noisy Scenes," *IEE Trans. Pattern Anal. & Mach. Intell.*, vol. 3, no. 1, pp. 103-111, Jan. 1981.

[106]J.F. Canny, "Finding Edges and Lines in Images," MIT AI-TR-720, June 1983.

[107]M. Miyahara, "Improvements of Television Picture Quality by Eliminating the Disturbances Caused by Interlaced Scanning," *IEEE Trans. Commun.*, vol. 31, no. 7, pp. 902-906, July 1983.

[108]Y. Ninomiya, Y. Ohtsuka, and Y. Izumi, "A Single Channel HDTV Broadcast System - The Muse," *NHK Lab. Note*, no. 304, pp. 1-12, Sep. 1984.

[109]H. Schroder, M. Silverberg, B. Wendland, and G. Huerkamp, "Scanning Modes for Flicker-Free Colour TV Reproduction," *IEEE Trans. Consumer Electronics*, vol. 31, no. 4, pp. 627-641, Nov. 1985.

[110]D. Teichner, "Quality Improvement by Adaptive Inter-/Intraframe Processing in PAL TV Receivers," *IEEE Trans. Cons. Elect.*, vol. 31, no. 3, pp. 226-239, Aug. 1985.

[111]A. Roberts, "The Improved Display of 625-Line Television Pictures: Adaptive Interpolation," BBC RD 1985/5, May 1985.

[112]D. Gabor and P.C.J. Hill, "Television Band Compression by Contour Interpolation," *Proc. Inst. Elec. Eng*, vol. 108, pp. 303-315, May 1961.

[113]A.R. Billings and A.B. Scolaro, "The Gabor Compression-Expansion System Using Non-Gaussian Windows and Its Application to Television Coding and Decoding," *IEEE Trans. Inf. Theory*, vol. 22, no. 2, pp. 174-507, Mar. 1976.

[114]H.S. Hou and H.C. Andrews, "Cubic Splines for Image Interpolation and Digital Filtering," *IEEE Trans. Acoust., Speech, Sig. Process.*, vol. 26, no. 6, pp. 508-517, Dec. 1978.

[115]Y. Nakajima, Y. Mizutami, S. Tamaki, H. Ito, T. Kasezawa, and T. Murakami, "Improvement of Picture Quality for NTSC and PAL Systems by Digital Signal Processing," *IEEE Trans. Consumer Electronics*, vol. 31, no. 4, pp. 642-654, Nov. 1985.

[116]L.J. Thorpe, T. Nakamura, and K. Ninomiya, "Super Motion System," *SMPTE J.*, vol. 94, no. 9, pp. 896-903, Sept. 1985.

[117]S.C. Hsu, "The Kell Factor: Past and Present," *SMPTE J.*, vol. 95, no. 2, pp. 206-214, Feb. 1986.

[118]H.P. Lavin, "System Analysis" in *Photoelectric Imaging Devices, Vol. I*, ed. L.M. Biberman, Plenum Press, 1971, pp. 333-375.

[119]J. Ward, "Wobbled-Raster High-Resolution Display," MAC-MEMO-431, MIT Electronic Systems Lab., Mar. 1974.

[120]*The Focal Encyclopedia of Film and Television Technology,* Focal Press Ltd., 1969.

Appendix A

Documentation for TV Simulation Programs

This section contains UNIX manual pages for important programs used in this research. To gain a full understanding of each program, read the relevant chapters in the text.

The programs are documented in the following order:

Chapter 2:

erase - simulates electron beam erasure.

Chapter 3:

pel - simulates dynamics of a single camera and display pixel.

int - temporally integrates frames of an image sequence.

rec - reconstructs image display function from video signal.

tip - integrates a 3-D analytic image illuminance function.

Chapter 4:

edge - finds orientation and magnitude of edges in image datfiles.

svfilt - performs linear space-variant filtering on a 2-D datfile.

advint - performs adaptive 2:1 vertical line interpolation.

Chapter 5:

tvsim - TV simulation program.

ar_crop - crops input picture to correct aspect ratio.

lens - simulates magnification and filtering of camera lens.

camera - simulates tube-type or solid-state video camera.

prefilt - simulates digital or analog video prefilter.

channel - simulates noisy analog video channel.

postfilt - simulates digital or analog video postfilter.

display - simulates tube-type or solid-state video display.

tvpars - prints out TV simulation parameters.

NAME

erase – simulates electron beam charge erasure

SYNOPSIS

erase in filt out [-m mode] [-g gain] [-w N] [-v]

DESCRIPTION

erase simulates destructive readout of a charge image by an electron beam, producing a single line of video information. For more information, see Chapter 2 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in and *filt* are 2-D datfiles. They may be of any data type. *in* is the charge image, and *filt* is the current distribution in the electron beam. The first V lines of *in* are scanned by the beam, where V is the vertical size of the beam. As the beam moves to the right one pixel at a time, the amount of charge neutralized is summed to form an output point in the video signal. *out* is the 1-D video signal, of the same data type as *in*, unless the –w N option is present, in which case *out* is the 2-D modified charge image at the Nth time step.

The following options are recognized:

–m mode

This is the readout mode. If *mode=0*, passive linear filtering is performed. This is the default mode. No charge is destroyed, and a simple linear convolution formula is used to compute the video signal. If *mode=1*, linear charge erasure is performed. Assume there are Q units of charge at a certain location just before readout, and there are B units of charge delivered by the beam at the same location. If Q≥B, then B units of charge are read out, and Q–B remain. If Q<B, then Q units are read out, and zero remain. All the Q's are summed to form the readout signal. If *mode=2*, exponential charge erasure is performed. Each beam value is interpreted as the *fraction* of charge to be read out; therefore, the beam should contain values in the range [0,1]. Q*B charge units are read out, and Q*(1–B) remain. All the Q's are summed to form the readout signal.

–g gain

gain multiplies all the beam values before erasure. This option may be used to vary the beam current without having to generate new beam filters. It may also be used to normalize the beam filter so that all values are in the range [0,1] (see mode 2, above).

–w N If this option is specified, the output signal is the modified charge image at the Nth time step. N must be in the range [0,H–1], inclusive, where H is the horizontal size of the charge image.

–v The program prints out the values of all the parameters.

EXAMPLE

Assume **test** is a datfile of size 256H×19V, and **filt** is a Gaussian filter of size 19×19. To perform linear erasure on **test**, producing the video signal **video**, **type**

erase test filt video -m 1

SEE ALSO

view(l), tvsim(l), camera(l)

DIAGNOSTICS

Error checking is minimal.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 11, 1986.

NAME

pel – simulates dynamics of a single camera and display pixel

SYNOPSIS

pel in out [options]

DESCRIPTION

pel simulates the dynamics of light-to-charge conversion in a single camera element and the subsequent charge-to-light conversion in the corresponding display element. For more information, see Chapter 3 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 1-D datfile input, which is of any data type, but should be of type *float* or *double* to produce the most accurate results. *in* contains N points of the luminous flux waveform; therefore each value should be nonnegative. This waveform is converted to charge, integrated, and read out by the camera element; the video signal is then converted back to light by a simulated phosphor display element. *out* is the name of the output datfile. It has the same data type as *in*, and contains *five* datasets. Each dataset is N points long and represents a different waveform in the 1-D simulation:

Dataset 0 is the same as *in*, and is included for plotting purposes.

Dataset 1 is the photocurrent waveform. Saturation is not modeled, so the photocurrent waveform should, in general, follow changes in the luminous flux. Parameters that affect the photocurrent waveform are G, the photoconductive gain; G1, the photoconductive gamma; TL, the time constant of photoconductive lag; ID, the dark current; and B, the bias light current.

Dataset 2 is the waveform of charge integration and readout for the camera element. The photocurrent is integrated over at most N/NF points, where N is the number of data points and NF is the number of frames. Other parameters that affect this waveform are P, the percent of the frame period for integration; R, the readout parameter; and **–b**, the signal-dependent beam lag option.

Dataset 3 is the readout signal for the single camera element. It is nonzero only at multiples of the frame period because the camera element is sampled only once per frame. Parameters that affect this waveform are R, the readout parameter; and **–b**, the signal-dependent beam lag option.

Dataset 4 is the display waveform. The readout signal is optionally gamma-corrected and upsampled. It is then converted to light by simulating the exponential decay of a typical CRT phosphor. Parameters that affect this waveform are G2, the external gamma; G3, the display gamma; U, the upsampling rate at the display; and TD, the time constant of phosphor decay.

The following options are recognized:

–n NF NF is the number of frame periods in the simulation. The default is 30, which means the simulation covers 1 second in an NTSC system. The number of points per frame period is N/NF, where N is the total number of points in the input file. Usually, 30 to 50 points per frame period provides adequate temporal resolution. Thus, for a 30-frame simulation, the input datfile should contain 900 to 1500 points.

–p P P determines the percent of frame period for integration. The default is 100%, which means all photocharges are integrated before readout. Thus, P controls the *integration time* of the camera element.

–t TL TD

Light-to-charge conversion is not instantaneous in a video camera. The photocurrent lags the illuminance function in time. In this program, the response to a step change of illumination is modeled as an exponential characterized by a single time constant. TL is the

time constant of photoconductive lag in frame periods. The default is 0, implying no lag. In tube-type displays, the impulse response of charge-to-light conversion is modeled as a decaying exponential characterized by a single time constant. TD is the time constant of phosphor decay in frame periods. The default is 0, implying a stroboscopic display. If the **-t** option is present, both TL and TD must be specified.

-r R R affects the magnitude and type of charge readout. It may be interpreted as a relative or absolute quantity. Assume there are Q units of integrated charge just before readout. If $0 \leq R \leq 1$, then the R*Q charge units are read out at that location, and (1–R)*Q charge units remain. If $R>1$, then Q units are read out and 0 remain if $R \geq Q$, or R units are read out and Q–R remain if $R<Q$. The default value is 1.0, implying 100% readout. The **-b** option can be used in conjunction with a relative readout efficiency to model signal-dependent beam lag.

-i ID *ID* is the dark current. This value is added to the photocurrent after gain and gamma conversion but before lag. The default is 0.

-g G *G* is the photoconductive gain factor. It is applied to each value of gamma-converted illuminance. The default is 1.0.

-G G1 G2 G3
G1 is the photoconductive gamma that is applied to each value of input illuminance in the light-to-charge conversion process. G2 is the external gamma, applied to the readout signal just before it is sent to the display. G3 is the display gamma, applied to the upsampled video signal before filtered by the display impulse response. If the **-G** option is present, all parameters must be specified, otherwise all defaults are 1.0.

-b B B is the bias photocurrent that is added to the original photocurrent. A proportional amount is subtracted from the readout signal to preserve the average level. This option should be used in conjunction with a *relative* readout efficiency to model signal-dependent beam lag. When the **-b** option is present, the readout efficiency is equal to 0.9 for $Q>5000$, where Q is the amount of integrated charge just before readout. The readout efficiency linearly decreases to R, the specified fraction ($0<R<0.9$), as Q approaches zero. Thus, at lower signal levels, the readout efficiency is worse, implying longer lag. This signal-dependent lag is observed in practice.

-u U U is the upsampling rate at the display. The video signal is linearly interpolated by an integral factor U. This increases the refresh rate of the display, and decreases the magnitude of flicker.

-v The program prints out the values of all the parameters.

-V The program prints out the values of all the parameters and some processing information.

EXAMPLE

To simulate a TV system with a linear lag-free photoconductor, an integration time equal to half the frame period, a readout efficiency of 90%, and external gamma of 0.65, a display gamma of 2.2, and a phosphor decay time equal to 1% of the frame time, type

pel in out -p 50 -G 1 0.65 2.2 -t 0 0.01 -r 0.9

SEE ALSO

dgraph(l), int(l), rec(l), tip(l)

DIAGNOSTICS

Error checking is minimal. The user must insure that the input file is nonnegative.

AUTHOR

Michael A. Isnardi

NAME

int – temporally integrates frames of an image sequence

SYNOPSIS

int in out video [options]

DESCRIPTION

int temporally integrates a sequence of subframes, in much the same way a video camera integrates the image illuminance function on the camera target. For more information, see Chapter 3 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the datfile input sequence, which is of any data type. The size of each subframe must be given by the DIMENSIONS keyword, and the number of subframes must be given by the DATASETS keyword. Datasets are called subframes because several of them are needed to produce a single output frame of video. Subframes of *in* are converted to charge, integrated, and read out. Each dataset of *out*, which has the same data type as *in*, should be interpreted as the *charge remaining* on the camera target at the end of each subframe interval. Each dataset of *video*, which has the same data type as *in*, should be interpreted as the camera video signal.

The following options are recognized:

–s S *S* is the number of horizontal scans per subframe. It must be an integer in the range 1 (the default) to V, where V is the number of lines in each subframe. The larger the value, the faster the scan rate. In each subframe of charge, *S* lines of charge are scanned and read out, forming the lines of the video signal. The number of *full frames* in the sequence is S*N/V, where N is the number of subframes.

–p P P determines the percent of frame period for integration. The default is 100%, which means that at each location, the charge values from all subframes are integrated before readout. If P=50, only the first half of the subframes are used for integration. Thus, P controls the *integration time* of the video camera.

–t TL TR

Light-to-charge conversion is not instantaneous in a video camera. The photocurrent lags the illuminance function in time. In this program, the response to a step change of illumination is modeled as an exponential characterized by a single time constant. *TL* is the time constant of photoconductive lag in full frame times. The default is 0, implying no lag. Another factor that affects the charge on a camera target is lateral charge spreading, or relaxation. If charge spreads too much between scan intervals, image detail is lost. *TR* is the time constant of charge spreading in full frame times. The default is 1e6, implying no charge spreading. If the –t option is specified, both *TL* and *TR* must be specified.

–r R R affects the magnitude and type of charge readout. It may be interpreted as a relative or absolute quantity. Assume there are Q units of integrated charge at some location just before readout. If $0 \le R \le 1$, then the R*Q charge units are read out at that location, and (1–R)*Q charge units remain. If $R > 1$, then Q units are read out and 0 remain if $R \ge Q$, or R units are read out and Q-R remain if $R < Q$. The default value is 1.0, implying 100% readout.

–i ID *ID* is the dark current. This value is added to the photocurrent after gain and gamma conversion but before lag. The default is 0.

–I IR *IR* is the scanning interlace ratio. The default is 1, implying progressive scan. Any positive integer that divides evenly into the number of lines per subframe may be used.

–g G *G* is the photoconductive gain factor. It is applied to each value of gamma-converted

illuminance. The default is 1.0.

-G G1 G2
G1 is the photoconductive gamma that is applied to each value of input illuminance in the light-to-charge conversion process. *G2* is the external gamma, applied to the video signal just before it is written. If the -G option is present, both parameters must be specified, otherwise the default for both are 1.0.

-d This option converts *out* and *video* into 2-D files that may be displayed as a filmstrip on a monitor using *view(l)*.

-z This option forces zero initial conditions in the computation of lag and integration. If not specified, the -1-th subframe of charge is identical to the zero-th subframe.

-v The program prints out the values of all the parameters.

-V The program prints out the values of all the parameters and a dynamic line count.

EXAMPLE

Assume **seq** is a datfile with 16 datasets, each of size 16×16. To simulate an ideal linear lag-free camera with 50% integration time, 2 scans per subframe (progressive), and 100% readout, type

int seq seq.out seq.video -s 2 -p 50

SEE ALSO

view(l), rec(l), tip(l)

DIAGNOSTICS

Error checking is minimal.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 11, 1986.

NAME

rec – reconstructs image display function from video signal

SYNOPSIS

rec in out [options]

DESCRIPTION

rec temporally reconstructs a sequence of video frames, in much the same way a video display reconstructs the video signal on the display screen. This program is normally used to reconstruct the video frames generated by *int(l)*. For more information, see Chapter 3 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the datfile video sequence, which is of any data type. The size of each frame must be given by the DIMENSIONS keyword, and the number of frames must be given by the DATASETS keyword. Each video line in *in* is converted to light by scanning onto a simulated phosphor screen. Each dataset of *out*, which has the same data type as *in*, should be interpreted as "snapshots" of the display screen at equally spaced intervals within the frame period. Each output dataset is a subframe, and the number of subframes per frame is determined by the number of scans per subframe.

The following options are recognized:

–s S S is the number of horizontal scans per output subframe. It must be an integer in the range 1 (the default) to V, where V is the number of lines in each input frame. The larger the value, the faster the scan rate. In each output subframe, S video lines are converted to light and scanned onto a simulated phosphor screen. The total number of output subframes (datasets) is N*V/S, where N is the number of input video frames.

–t TL TR
TL is the time constant of phosphor decay in full frame times. The default is 0, implying a stroboscopic display. TR is the time constant of lateral light spreading in full frame times. The default is 1e6, implying no light spreading. If the –t option is specified, both TL and TR must be specified.

–I IR IR is the scanning interlace ratio. The default is 1, implying progressive scan. Any positive integer that divides evenly into the number of lines per video frame may be used.

–G GD GD is the display gamma, the exponent of the charge-to-light transfer characteristic. The default is 1.0, but most CRT's have gammas in the range 2.2 to 2.8.

–d This option converts *out* into a 2-D datfile so that it may be displayed as a filmstrip on a monitor using *view(l)*.

–v The program prints out the values of all the parameters.

–V The program prints out the values of all the parameters, the interlace order, and the some processing information.

EXAMPLE

Assume **video** is a datfile with 16 datasets, each of size 16×16. To simulate a CRT with decay time constant of 0.01 frame period, 2 scans per subframe (progressive), and gamma of 2.2, type

rec video video.out -s 2 -t 0.01 0.0 -G 2.2

SEE ALSO

view(l), int(l), tip(l)

AUTHOR

Michael A. Isnardi

NAME

tip – integrates a 3-D analytic image illuminance function

SYNOPSIS

tip out [options]

DESCRIPTION

tip simulates the integration of light-converted charge for each element on the camera surface. The integration time offset for each pixel may be different, and a graphic representation of the integration periods at each spatial location is given by a *temporal integration pattern (TIP)* for the imaging device. Several common TIP's are simulated. For more information, see Chapter 3 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

out is the name of the output datfile. It has data type *unsigned_1*, unless the **-f** option is present, in which case the output is type *float*. Several options are used to generate an analytic image illuminance function that is integrated temporally at each location. The following options are recognized:

-d H V F
H and V are the desired horizontal and vertical dimensions of each output frame, and T is the desired number of frames (datasets). The defaults are 16×16×16.

-t TI TI is the fraction of each frame period for integration. The default is 1.0. Smaller fractions produce less motion blur.

-o obj This parameter picks the type of object and motion desired:

0: rectangle, constant motion (default).

1: ellipse, constant motion.

2: ellipse, sinusoidal motion.

3: ellipse, revolving motion.

4: 4 ellipses, revolving motion.

5: rotating rectangle.

-s x0 y0
x0 and y0 are the x (horiz) and y (vert) relative initial position of the center of the object in the frame. The defaults are (0.5, 0.5)

-v vx vy
vx and vy are the x and y initial velocities of the center of the object in pels per frame. The defaults are (0.0, 0.0).

-D h w h and w are the height (default 4) and width (default 4) of the object.

-b o b o is the object brightness (default 255) and b is the background brightness (default 0).

-n NT NS
NT is the number of subframe temporal increments (default 1) and NS is the number of subpixel spatial increments (default 1). To approximate the temporal integration, NT samples are summed at equally spaced increments within the integration interval. To approximate spatial integration within each element, the spatial area is divided into NS*NS subpixels. Usually, NS=1 is adequate. For small values of TI and object velocity, small values of NT are adequate. However, for large values of TI and/or object velocities, larger values, on the order of 10 or 30, are required to avoid contouring. Of course, the processing time increases as NT*NS*NS, so accuracy versus run time is a tradeoff.

-p per For object motion that is cyclic (modes 2,3,4,5), *per* determines the period of the cycle, in frame times. The default is 1.0.

-i IR IR is the interlace ratio, which affects the shape of the TIP. The default is 1 (progressive scan).

-m mode

mode selects the TIP:

0: movie film of progressive frame-transfer CCD. All pixels have the same time offset, independent of interlace. This is the default TIP.

1: line transfer CCD. All pixels along a line have the same offset; however, with progressive scan, the offset increases by 1/V frame periods from one line to the next, where V is the number of lines per frame. Interlace affects this mode.

2: tube-type camera. All pixels have a different time offset; along a line, each pixel starts integrating 1/H*V frame periods after its neighbor to the left, where H*V is the number of pixels per frame. Interlace affects this mode.

3: field-transfer CCD. All pixels in the same field have the same time offset. The interlace ratio determines the number of fields.

4: interlaced frame-transfer CCD. All pixels have the same time offset. However, there are IR fields per frame, where IR is the interlace ratio. At each horizontal location, IR pixels are spatially averaged to produce an output point, but the group of pixels shifts down by one pixel per field. For example, when IR=2, lines 1 and 2 are averaged to produce line 1 of the first field, and lines 2 and 3 are averaged to produce line 1 of the second field.

-f This option forces the output to be of type float.

-d This option converts the output to a 2-D file so that a datfile containing multiple frames may be viewed as a filmstrip on a monitor.

-V The program prints out the values of all the parameters, some processing information, and a dynamic line count.

EXAMPLE

Suppose we want a sequence **out** having 4 frames, each of size 256×256. To simulate a the motion rendition of a moving circle (diameter = 40 pixels, moving rightward 5 pixels per frame) using a 2:1-interlaced tube-type camera with integration time equal to half the frame period, type

tip out -d 256 256 4 -t 0.5 -o 1 -v 5 0 -D 40 40 -n 5 1 -i 2 -m 2

The number of temporal sampling increments is 5.

SEE ALSO

view(l), int(l), rec(l), pel(l)

DIAGNOSTICS

Error checking is minimal.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 14, 1986.

NAME

edge – finds orientation and magnitude of edges in image datfiles

SYNOPSIS

edge in out [–m mode] [–v]

DESCRIPTION

edge uses a 3×3 neighborhood computation to find the orientation and magnitude of local edges. For detailed description of the algorithms, see Chapter 4 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported. If only the first two arguments are specified, the program computes edge magnitude using the slope average formula.

The following options are recognized, and may appear in any order.

–m mode

mode determines the algorithm. If *mode = 0*, edge orientation is computed; the output is in degrees. If *mode = 1*, edge magnitude using slope average is computed. This is the default. If *mode = 2*, gradient magnitude is computed. If *mode = 3*, the angle between the surface normal and the z-axis is computed in degrees. This metric is also an indication of edge magnitude, but is more sensitive to noise.

–v If this *verbose* option is specified, the program will print out a dynamic count of the number of output lines written.

EXAMPLE

To compute the edge orientation of cman (in degrees), type

edge cman cman.out -m 0

SEE ALSO

svfilt(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 10, 1986.

NAME

svfilt – performs linear space-variant filtering on a 2-D datfile

SYNOPSIS

svfilt in out filtset address [–f] [–v –V]

DESCRIPTION

svfilt performs linear space-variant filtering on a 2-D datfile. The input file is interpreted as an array of impulses. Each impulse generates an impulse response that depends on its location, and the output file is the sum of all the impulse responses. For more information, see Chapter 4 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, which is of any data type, and *out* is the name of the desired output picture, which is of the same type as *in*, unless the **–f** option is specified, in which case the output is of type *float*.

filtset is a datfile containing a set of 2-D filters, of any data type. The dimensions of each 2-D filter must be given by the DIMENSIONS keyword; the number of 2-D filters must be given by the DATASETS keyword.

address is a 2-D datfile of any data type having the same dimensions as *in*. The values in *address* should be nonnegative integers in the range [0,N-1] inclusive, where N is the number of 2-D filters in the filter set. The value at each location in *address* picks the filter to use as an impulse response at the same location in the output image. The impulse responses are centered at each location. Each impulse response is weighted by the appropriate value of the input image, and all the impulse responses are summed to form the output image. If all the values in *address* are the same, then the same filter is always chosen; this reduces the filtering operation to a simple convolution.

The following options are recognized:

–f The output daftile if forced to be of type *float*. This option is useful for normalization. Normalizing the output image removes the gain artifacts in areas where filters change abruptly from point to point. To normalize, create an all-zero datfile *norm* and then insert 1's at all locations where the input image is non-zero. Filter the original and *norm* with the same *filtset* and *address*, using the **–f** option. Then divide the filtered original by the filtered *norm* datfile using *div(l)*. Finally, use *preview(l)* to view the result.

–v The program prints out a dynamic line count.

–V The program prints out the current location, the input value, and the address value. This is only done for nonzero input values. The program runs much slower when this option is specified, so it should only be used as a diagnostic.

EXAMPLE

To filter all the even lines of **cman** with **filt0** and all the odd lines with **filt1**, first create a datfile **filtset** with **filt0** as the first dataset and **filt1** as the second. The dimensions of **filt0** and **filt1** must be the same. Then create a 2-D datfile **address** with 0's on the even lines and 1's on the odd lines. The dimensions of **address** and **cman** must be the same. Then type

svfilt cman cman.sv filtset address

SEE ALSO

filter(l), preview(l), div(l)

DIAGNOSTICS

Error checking is minimal. The user must insure that all addresses are valid.

AUTHOR

Michael A. Isnardi

NAME

advint – performs adaptive 2:1 vertical line interpolation

SYNOPSIS

advint in out [B [R [S [V]]]]

DESCRIPTION

advint uses a block-matching scheme to interpolate contours on synthetic scan lines. A useful application is to convert a single television field into an entire frame. For a detail description of the algorithm, see Chapter 4 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, which is of any data type, and *out* is the name of the desired output picture, which is of the same type as *in*. The horizontal dimension of *out* and *in* are the same, but the vertical dimension of *out* is twice that of *in*.

The following options are recognized:

B This is the block size. The default value is 7. Blocks move in opposite directions on adjacent lines. The range parameter determines the maximum offset in either direction. At each offset, a block difference is computed, and the minimum block difference determines the best offset as well as the values to average for the interpolated pixel.

R This is the range parameter. The default is 1. The blocks move a maximum offset of *R* pixels in either direction. Large values of *R* may pick up detail that is not spatially coherent, providing an incorrect match.

S This is the subpixel factor. The default is 1. S-1 pixels are linearly interpolated between existing pixels. The algorithm is applied to the horizontally expanded image, and the result is subsampled by S.

V If *V=0*, no information is printed. If *V=1*, a dynamic line count is printed to *stderr*. If *V=2*, a dynamic line count is printed to *stderr* and the optimal offset at each location is printed to *stdout*. This information may be used to study the performance of the block matching algorithm.

The block size and range parameters affect the quality of interpolation; bad values may introduce artifacts. The default values produce good results on most images. The program can also perform 2:1 2-D interpolation on an original image. First, interpolate vertically using *advint*, then transpose the image and run *advint* again. Transpose a second time to match the format of the original. 4:1 interpolation can be performed by repeating this procedure.

EXAMPLES

To interpolate **cman.sub**, a vertically subsampled version of **cman**, with B=5 and R=2, type

advint cman.sub cman.adv 5 3

To print the offset values to the file **offset**, using parameters B=3, R=1, type

advint cman.sub cman.adv 3 1 2 > offset

SEE ALSO

filter(l), transpose(l)

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 10, 1986.

NAME

tvsim – TV simulation program

SYNOPSIS

tvsim pic [–e [par]] [–r [par]]

DESCRIPTION

tvsim is an interactive program that simulates the spatial processing of video cameras and displays. It can be used as a tool for the design and analysis of monochrome baseband TV systems. Programmed for generality, *tvsim* has an extensive set of parameters under user control. For a detailed description of the program, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

pic is the 2-D datfile input picture, which may be of any data type. The name *pic* is also used as the root name for processed pictures. For instance, the output of the camera subprogram is called *pic.c*. The intermediate processed pictures may have different dimensions from those of the *pic*, but they will have the same data type.

tvsim is a master program that reads in a set of TV parameters and then executes a sequence of subprograms to process the input picture. The subprograms are executed in the following order:

ar_crop → *lens* → *camera* → *prefilt* → *channel* → *postfilt* → *display*

Depending on the parameters, some subprograms may be omitted from this sequence. Read the manual page for each subprogram to find the relevant parameters and their effect. Each subprogram may be run as in independent process.

tvsim runs in one of three possible modes: design, analysis, and rerun.

Design mode is the default mode, and is entered by typing the command

tvsim pic

where pic is the name of the input image datfile. In this mode, one begins by specifying the TV components (subprograms) to be included in the simulation. Then the parameters for each component are interactively specified. After this, one may edit the parameter list to correct any errors. The *tvsim* parameters are converted to datfile keywords, appended onto the descriptor portion of *pic*, and processing begins. Analysis, or edit, mode is entered by typing the command

tvsim *pic* -e [pars]

where *pars* is an optional ordinary file containing a predefined set of *tvsim* parameters. If *pars* is not specified, parameters are taken from the descriptor portion of *pic*, which implies that *pic* should have been run through *tvsim* at least once in design mode. After the parameters are edited, processing begins. The effect of varying a single parameter, or small set of parameters, may be investigated using this mode.

Rerun mode is entered by typing the command

tvsim *pic* -r [pars]

where *pars* is an optional ordinary file containing a predefined set of *tvsim* parameters. Processing is immediately carried out using the parameters in the parameter file *pars* (if specified), or in the descriptor portion of *pic*. This mode is useful for regenerating a processed image without having to respecify the parameters.

SEE ALSO

ar_crop(l), lens(l), camera(l), prefilt(l), channel(l), postfilt(l), display(l), tvpars(l)

AUTHOR

Michael A. Isnardi

NAME

ar_crop – crops input picture to correct aspect ratio

SYNOPSIS

ar_crop in out [AR]

DESCRIPTION

ar_crop crops the input picture to the largest size with the desired aspect ratio. The output picture is centered in the dimension of the crop. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

If *AR* is present, it is used as the aspect ratio (the ratio of width to height). The picture is cropped to the largest size with the desired aspect ratio. This implies that cropping occurs in only one dimension. The output image is centered along the cropped dimension.

If only the first two arguments are specified, the program reads the value of the keyword *c_ar* from the descriptor portion of *in*, and uses that value as the aspect ratio. This is what happens when *ar_crop* is called from *tvsim(l)*. In this case, the output picture is called *in.ar*.

EXAMPLE

To crop the datfile **pic** to a 4:3 aspect ratio, type

ar_crop pic pic.ar 1.333

SEE ALSO

tvsim(l), camera(l), lens(l), crop(l)

DIAGNOSTICS

Large picture sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 14, 1986.

NAME

lens – simulates magnification and filtering of camera lens

SYNOPSIS

lens in out

DESCRIPTION

lens simulates the spatial scaling and filtering properties of space-invariant lenses. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

c_l (1) :

upsampling factor. The image is upsampled by this amount.

c_m (1) :

downsampling factor. The image is downsampled by this amount. The overall magnification is *c_l/c_m*.

c_lens (1) :

lens filter type, or point spread function (PSF). If *c_lens=0*, a separable Gaussian of size 4*SCALE–1×4*SCALE–1 is used, where SCALE is max*(c_l,c_m)*. If *c_lens=1*, a separable bilinear filter of size 2*SCALE–1×2*SCALE–1 is used. If *c_lens=2*, a separable boxcar of size SCALE×SCALE is used. If *c_lens=6*, a 1×1 impulse is used.

verbose (1):

If *verbose=1*, filter coefficients and a dynamic line count are printed.

If spatial scaling is desired in addition to filtering, the input image is first zero-padded by a factor of *c_l* in both dimensions, filtered with the desired PSF, and then subsampled by a factor of *c_m* in both dimensions.

SEE ALSO

tvsim(l), camera(l), ar_crop(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 14, 1986.

NAME

camera – simulates tube-type or solid-state video camera

SYNOPSIS

camera in out

DESCRIPTION

camera simulates the light-to-charge and scanning properties of tube-type and solid-state video cameras. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

c_proc (1):
camera processing flag. Processing is performed only if *c_proc=1*.

c_vaper (5):
vertical aperture size.

c_haper (5):
horizontal aperture size.

c_vpitch (4):
vertical pitch.

c_hpitch (1):
horizontal pitch. If the input picture is of size H×V, the output picture will be of size H/c_hpitch×V/c_vpitch.

c_aper (0):
aperture type. The options are Gaussian (0), bilinear (1), prism (2), sinc(x) (3), |sinc(x)| (4), sinc(x)*sinc(x) (5), impulse (6), cylinder (7), cos(x)*cos(x) (8).

c_ir (1):
interlace ratio. The default is 1, progressive scan.

c_erase (0):
charge erasure mode. Three modes are available: passive linear filtering (0), linear erasure (1), and exponential erasure (2).

c_amp (100.0):
amplitude of camera aperture. This parameter affects charge erasure.

c_gam1 (1.0):
photoconductive gamma. For vidicons, *gam1≈0.65*. For Plumbicons and most CCD's, *gam1≈1.0*.

c_gam2 (1.0):
external gamma correction.

c_vsig (1.0):
Gaussian vertical sigma.

c_hsig (1.0):
Gaussian horizontal sigma. Sigma values are relevant for Gaussian apertures only.

c_apname (null):
name of optional datfile filter for camera aperture.

verbose (1):
If *verbose=1*, filter coefficients and a dynamic line count are printed.

SEE ALSO
tvsim(l), lens(l), ar_crop(l), filter(l)

DIAGNOSTICS
Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR
Michael A. Isnardi

DATE
Revised Apr. 15, 1986.

NAME

prefilt – simulates digital or analog video prefilter

SYNOPSIS

prefilt in out

DESCRIPTION

prefilt spatially scales its input in one or more dimensions. It can be used as a standards converter, a 2-D aperture corrector, or as a sample-and-hold filter for a solid-state camera. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

pr_proc (1):
prefilter processing flag. Processing is performed only if *pr_proc=1*.

pr_filt (2):
filter type. The options are Gaussian (0), bilinear (1), prism (2), sinc(x) (3), |sinc(x)| (4), sinc(x)*sinc(x) (5), impulse (6), cylinder (7), cos(x)*cos(x) (8), 3×3 sharpening filter (11).

pr_vsize (2):
vertical filter size.

pr_hsize (2):
horizontal filter size.

pr_vl (1):
vertical upsampling factor.

pr_vm (2):
vertical downsampling factor.

pr_hl (1):
horizontal upsampling factor.

pr_hm (2):
horizontal downsampling factor. If the input picture is of size H×V, the output picture will be of size H**pr_hl/pr_hm*× V**pr_vl/pr_vm*.

pr_amp (100.0):
filter amplitude.

pr_name (null):
name of optional datfile filter for prefilter.

verbose (1):
If *verbose=1*, filter coefficients and a dynamic line count are printed.

SEE ALSO

tvsim(l), camera(l), channel(l), postfilt(l), display(l), filter(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

NAME

channel – simulates noisy analog video channel

SYNOPSIS

channel in out

DESCRIPTION

channel filters and adds common degradations to the input picture, which is assumed to be a video signal. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same dimensions and data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

ch_proc (1):
prefilter processing flag. Processing is performed only if *ch_proc=1*.

ch_filt (6):
1-D lowpass horizontal filter type. The options are Gaussian (0), triangle (1), boxcar (2), sinc(x) (3), |sinc(x)| (4), sinc(x)*sinc(x) (5), impulse (6). Filter size depends on the filter chosen.

ch_cutoff (1):
channel normalized cutoff frequency (1 => π, 0.5 => 0.5π, etc.). The specified filter will have approximately this cutoff frequency.

ch_ghdel (0):
ghosting delay in pixels.

ch_ghamp (0):
ghosting amplitude.

ch_snr (-1):
channel SNR in dB with additive white Gaussian noise (-1 => none).

verbose (1):
If *verbose=1*, filter coefficients and a dynamic line count are printed.

SEE ALSO

tvsim(l), camera(l), channel(l), postfilt(l), display(l), filter(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 15, 1986.

NAME

postfilt – simulates digital or analog video postfilter

SYNOPSIS

postfilt in out

DESCRIPTION

postfilt spatially scales its input in one or more dimensions. It can be used as a standards converter, a 2-D channel inverse filter, or as an inverse filter for the display aperture. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

po_proc (1):
: postfilter processing flag. Processing is performed only if *po_proc=1*.

po_filt (1):
: filter type. The options are Gaussian (0), bilinear (1), prism (2), sinc(x) (3), |sinc(x)| (4), sinc(x)*sinc(x) (5), impulse (6), cylinder (7), cos(x)*cos(x) (8), 3×3 sharpening filter (11).

po_vsize (3):
: vertical filter size.

po_hsize (3):
: horizontal filter size.

po_vl (2):
: vertical upsampling factor.

po_vm (1):
: vertical downsampling factor.

po_hl (2):
: horizontal upsampling factor.

po_hm (1):
: horizontal downsampling factor. If the input picture is of size H×V, the output picture will be of size H**po_hl/po_hm*× V**po_vl/po_vm*.

po_amp (100.0):
: filter amplitude.

po_name (null):
: name of optional datfile filter for postfilter.

verbose (1):
: If *verbose=1*, filter coefficients and a dynamic line count are printed.

SEE ALSO

tvsim(l), camera(l), prefilt(l), channel(l), display(l), filter(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

NAME

display – simulates tube-type or solid-state video display

SYNOPSIS

display in out

DESCRIPTION

display simulates the charge-to-light and scanning properties of tube-type and solid-state video displays. This program is a standard subprogram in *tvsim(l)*, but may also be run as a separate program with the appropriate *tvsim* keywords. For more information, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

in is the 2-D datfile input picture, and *out* is the name of the desired output picture, which is of the same data type as *in*. All common data types are supported.

The program reads the values of the following *tvsim* keywords (default values are given in parentheses):

d_proc (1):
display processing flag. Processing is performed only if *d_proc=1*.

d_vaper (9):
vertical aperture size.

d_haper (9):
horizontal aperture size.

d_vpitch (4):
vertical pitch.

d_hpitch (1):
horizontal pitch. If the input picture is of size H×V, the output picture will be of size H**d_hpitch*×V**d_vpitch*.

d_aper (0):
aperture type. The options are Gaussian (0), bilinear (1), prism (2), sinc(x) (3), |sinc(x)| (4), sinc(x)*sinc(x) (5), impulse (6), cylinder (7), cos(x)*cos(x) (8).

d_ir (1):
interlace ratio. The default is 1, progressive scan.

d_amp (100.0):
amplitude of camera aperture. This parameter affects charge erasure.

d_gam (1.0):
display gamma. For most CRT's, *d_gam* is in the range 2.2 - 2.8.

d_vsig (1.5):
Gaussian vertical sigma.

d_hsig (1.5):
Gaussian horizontal sigma. Sigma values are relevant for Gaussian apertures only.

d_apname (null):
name of optional datfile filter for display aperture.

d_cont (4.0):
contrast scaling factor. For most aperture shapes, a good choice is *d_hpitch*d_vpitch*.

verbose (1):
If *verbose=1*, filter coefficients and a dynamic line count are printed.

SEE ALSO

tvsim(l), camera(l), filter(l)

DIAGNOSTICS

Large picture sizes and large vertical filter sizes can cause memory allocation failures on the PDP 11/45.

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 15, 1986.

NAME

tvpars – prints out TV simulation parameters

SYNOPSIS

tvsim pic

DESCRIPTION

tvpars prints to stdout the *tvsim* parameters found in the descriptor portion of *pic*. For a detailed description of the program and parameters, see Chapter 5 of M. Isnardi's Ph.D. thesis, "Modeling the Television Process", (MIT Dept. of Electr. Eng., May, 1986).

pic is the 2-D datfile input picture, which may be of any data type.

tvpars is used by *tvsim(l)* to prints out the parameters in design or edit mode. However, it may also be used to print out the parameters of any datfile to *stdout*. A typical application is to redirect the output of *tvpars* into an ordinary file, and then use this file as a parameter file for *tvsim*. For instance, suppose the 2-D datfile **cman** was processed by *tvsim*. To save the TV parameters in the file **par**, type

tvpars cman > par

To process another 2-D datfile **lady** with the same parameters, type

tvsim lady -r par

This saves time because the user does not have to interactively specify the same set of parameters for subsequent pictures.

SEE ALSO

tvsim(l), ar_crop(l), lens(l), camera(l), prefilt(l), channel(l), postfilt(l), display(l)

DIAGNOSTICS

If no *tvsim* keywords are found, the program prints " *pic* is not a TVSIM datfile. Process with TVSIM."

AUTHOR

Michael A. Isnardi

DATE

Revised Apr. 14, 1986.

Appendix B

UNIX Datfiles

In the early 1980's, UNIX datfiles were developed in the Cognitive Information and Digital Signal Processing Groups at MIT as a way of organizing and manipulating digitized data. A UNIX datfile is a directory containing one or more files. One of these files, named *data*, contains the data organized in a rectangular matrix with any number of dimensions. The most common data types are supported by the standard, and provision is made for non-standard data types and data file structures. A separate file in the datfile directory, named *descriptor*, contains a pseudo-LISP description of the data file. Keyword/value pairs of ASCII strings, enclosed in matching parentheses, describe the organization, data type, history and other aspects of the datfile. Users need not worry about the exact format of this descriptor file, since a wide range of subroutines are available to read, parse, modify and write these descriptor files.

Several keywords are considered "standard", and are read and interpreted by all utility programs written using this standard. The most important keywords are TYPE, DATASETS, CHANNELS and DIMENSIONS. The TYPE keyword describes the type of the data file elements. Digitized images are usually of type *unsigned_1*, meaning one unsigned byte per pixel. The DATASETS, CHANNELS and DIMENSIONS keywords describe the organization and size of the data file. A data file is composed of a set of equal-sized "datasets", corresponding to separate sets of experimental data. The number of data sets in the file is specified by the DATASETS keyword. (If the DATASETS keyword is omitted, the file is assumed to contain 1 dataset.) Each data set may contain multiple "channels" of data, all of which have identical sizes. The number of channels in each dataset is specified by the CHANNELS keyword. (If this keyword is omitted, each dataset is assumed to contain 1 channel.) Finally, each "channel" may be a multidimensional array, whose dimensions are given by the DIMENSIONS keyword. Each array is stored in the usual C order, with the last index varying most rapidly (i.e. a two dimensional array is stored by rows.)

Images and image sequences can be organized quite naturally in the datfile standard. A single monochrome image is usually organized as a 2-D array whose size is specified by two positive integers in the DIMENSIONS keyword. For an image sequence, the DATASETS keyword indicates the number of frames. An RGB color image or image sequence is organized as three CHANNELS; each channel contains a single color component.

The program *tvsim* creates special keywords and appends them to the descriptor portion of the input image. These keywords, or TV parameters, are copied from input to output as the image is processed by each *tvsim* subprogram. Thus, each intermediate image has a copy of the set of TV parameters that was used to create it. The program *tvpars* neatly displays these parameters; it may also be used to write them to an ordinary file for subsequent processing by *tvsim*.

Appendix C

Modulation Depth of Gaussian Beam

Modulation depth is an important measure of a CRT's small-area contrast. The greater the modulation depth, the greater the contrast and apparent resolution. Two modulation depths can be defined for a raster-scanned display. The first is the modulation of a *flat field*, and the second is the modulation of a *single blanked line*.

Assume the display beam is modeled as a Gaussian. Its normalized luminance profile in the vertical direction is given by

$$L(y) = e^{-\frac{y^2}{2\sigma^2}} . \tag{C.1}$$

Let p be the vertical spacing between scan lines. If $\sigma << p$, detail is preserved, but the scan lines are very visible. As the ratio p/σ decreases, scan lines become less visible, but contrast of high-frequency information is decreased.

The effect of p/σ ratio on both types of modulation is shown in Fig. 5.3(a). Each curve represents the vertical luminance profile of eight successive scan lines at a different p/σ ratio. All video lines have the same amplitude except the sixth, which is blanked. A flat field condition is approached at $p/\sigma = 1.75$, or $\sigma = 0.57p$. At this ratio, however, the relative contrast between lines 4 and 6 is reduced.

Fig. 5.3(b) shows the luminance profile at $p/\sigma = 2.5$. The luminance values used to determine the modulation depths are L_{MAX}, the maximum luminance; L_{MIN_F}, the minimum luminance in the flat field portion of the profile; and L_{MIN_L}, the minimum luminance of the blanked line. The modulation depths for the flat field (MF) and for the single blanked line (ML) are defined as follows:

$$MF = \frac{L_{MAX} - L_{MIN\_F}}{L_{MAX} + L_{MIN\_F}} \times 100\% \tag{C.2}$$

$$ML = \frac{L_{MAX} - L_{MIN\_L}}{L_{MAX} + L_{MIN\_L}} \times 100\% \qquad \text{(C.3)}$$

The ratio p/σ may assume any positive value. For a Gaussian beam, L_{MAX}, L_{MIN_F}, and L_{MIN_L} can be computed as a function of p/σ:

$$L_{MAX} = \sum_{n=-N}^{N} L(np) = L(0) + 2\sum_{n=1}^{N} L(np) \qquad \text{(C.4)}$$

$$= 1 + 2\sum_{n=1}^{N} e^{-\frac{n^2}{2}\left(\frac{p}{\sigma}\right)^2}$$

$$L_{MIN\_L} = L_{MAX} - 1 \qquad \text{(C.5)}$$

$$L_{MIN\_F} = 2\sum_{n=0}^{N} L\left((2n+1)\frac{p}{2}\right) \qquad \text{(C.6)}$$

$$= 2\sum_{n=0}^{N} e^{-\frac{(2n+1)^2}{8}\left(\frac{p}{\sigma}\right)^2}$$

Eqs. (C.2) and (C.3) can then be evaluated to produce the modulation curves shown in Fig. 5.3(c). A value of $N = 35$ was used to generate these curves.