AFHRL-TR-90-25

EFFICIENT IMAGE GENERATION USING LOCALIZED FREQUENCY COMPONENTS MATCHED TO HUMAN VISION

George A. Geri

University of Dayton Research Institute
300 College Park Avenue
Dayton, Ohio 45469

Yehoshua Y. Zeevi

CAIP Center
Rutgers University
Piscataway, N.J. 08855-1390

Moshe Porat

Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa, 32000, Israel

OPERATIONS TRAINING DIVISION
Williams Air Force Base, Arizona 85240-6457

July 1990

Final Technical Report for Period October 1987 - February 1990

AIR FORCE

AD-A224 903

HUMAN RESOURCES

LABORATORY

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.

CLAIRE FITZPATRICK, Captain, USAF
Contract Monitor

DEL ANDREWS, Technical Director
Operations Training Division

HAROLD G. JENSEN, Colonel, USAF
Commander

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 1990	Final Report - October 1987 to February 1990

4. TITLE AND SUBTITLE

Efficient Image Generation Using Localized Frequency Components Matched to Human Vision

5. FUNDING NUMBERS

C - F33615-87-C-0012
PE - 62205F, 61102F
PR - 1121, 2313
TA - 03, T3
WU - 83, 12

6. AUTHOR(S)

George A. Geri
Yehoshua Y. Zeevi
Moshe Porat

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Univeristy of Dayton Research Institute
300 College Park Avenue
Dayton, Ohio 45469

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Operations Training Division
Air Force Human Resources Laboratory
Williams Air Force Base, Arizona 85240-6457

10. SPONSORING MONITORING AGENCY REPORT NUMBER

AFHRL-TR-90-25

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Following a brief tutorial in the general area of image analysis, a formalism is presented for using the generalized Gabor approach to image representation in the combined frequency-position space. This approach uses elementary functions to which the human visual system is particularly sensitive and which are efficient for the analysis and synthesis of visual imagery. Among the topics covered are the complementarity of position and spatial frequency in the Gabor scheme, and the use of an auxiliary function to render the nonorthogonal Gabor elementary functions transformable. The formalism is in particular compatible with the implementation of a variable resolution system wherein image information is nonuniformly distributed across the visual field in accordance with the human visual system's ability to process it. A possible hardware implementation of such a system is described and some potential problems associated with its development are discussed.

14. SUBJECT TERMS

basis functions
computer-generated imagery
flight sumulation
image analysis
image processing

15. NUMBER OF PAGES

68

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

SUMMARY

We begin this report with an introductory review of some important terms and concepts in the area of image analysis. Among the topics covered are one- and two-dimensional periodic functions, frequency representation of simple images, frequency bandwidth, orthogonality, basis properties, and Fourier analysis. We then present a formal method for representing images in the combined position-spatial frequency space. Specifically, a technique is described for generating visual images by adding together luminance distributions in the form of sinusoids confined by Gaussian envelopes. These distributions are known as Gabor elementary functions (GEFs). The main advantage in using GEFs to generate images is that they are spatially localized and hence provide an efficient means for representing images in which information is distributed nonuniformly. Thus, GEFs can be used to generate images which can be efficiently matched to the inhomogeneities of the human visual system. In addition, the method described here for representing images is shown to be compatible with a variable resolution scheme which distributes image information based on the human visual system's ability to use it. A possible hardware implementation of the present image generation technique is described and some potential problems associated with its development are discussed.

PREFACE

This research was performed in support of the Training Technology Planning Objective of the Research and Technology Plan at the Operations Training Division of the Air Force Human Resources Laboratory, Williams Air Force Base, Arizona. The general objective of this training research and development program is to identify and demonstrate cost-effective strategies and new training systems for developing and maintaining combat effectiveness. The purpose of the present experiment was to elucidate the basic mechanisms underlying visually guided behavior in flight simulators.

This research was supported by the Air Force Office of Scientific Research, Life Sciences Task 2313T3, Work Unit 2313-T3-12 entitled Cognitive Aspects of Flight Training (Principal Investigator, Dr. Elizabeth L. Martin), and by Work Unit 1123-03-83, Flying Training Research Support, Air Force Contract F33615-87-C-0012 (UDRI), Capt. Claire Fitzpatrick, Contract Monitor. Dr. Y.Y. Zeevi was on leave from the Technion-Israel Institute of Technology while performing the research described in this report.

TABLE OF CONTENTS

		Page
I.	GENERAL INTRODUCTION	1
II.	PERIODIC FUNCTIONS AND FOURIER THEORY	2
	Introduction	2
	One-Dimensional Periodic Functions: The Sinewave Grating	2
	Two-Dimensional Periodic Functions	7
	Frequency Representation of Simple Images	8
	Spatial Frequency Bandwidth	13
	Basis Properties, Orthogonality, and Other Characteristics of Sinusoids Which Make Them Useful for Representing Images	16
	Frequency Analysis and Synthesis of Complex Images (Fourier Theory)	17
	Relevance of Fourier Analysis to Vision	18
	Image Representation Using Nonorthogonal Bases	19
III.	THE GABOR SCHEME: IMAGE REPRESENTATION IN THE COMBINED POSITION-SPATIAL FREQUENCY SPACE	21
	Introduction	21
	Image Representation Using Gabor Functions	26
	Variable Resolution Using Nonuniform Gabor Sampling	31
	Synthesis of Fully Textured Images Using Gabor Functions	35
IV.	GENERAL DISCUSSION	39
	REFERENCES	40
	SOURCE CODE FOR PROGRAMS GABAN.F AND GANAL.F	43

LIST OF FIGURES

Figure		Page
1	Example of Random and Periodic Images	3
2	Parameters Which Characterize a Sinusoidal Grating	4
3	Examples of One- and Two-Dimensional Sinusoidal Gratings	5
4	Frequency Representation of Gratings Shown in Figure 3	9
5	Phase Representation in Frequency Space	10
6	Two-Dimensional Spatial-Frequency Space	12
7	A Grating Image as the Sum of an Infinite Grating and a Window Function	14
8	Dependence of Image Bandwidth on the Spatial Extent of the Window Function	15
9	Dependence of Perceived Coherence on Component Phase	20
10	Forward and Inverse Gabor Transforms	22
11	Examples of Gabor Functions and of an Auxiliary Function	25
12	Space/Frequency Trade-off and Examples of Various Tessellation Schemes	27
13	Reconstruction of a Signal From Its Gabor Components	29
14	Nonuniform Sampling and Tessellation and a Resulting Variable Resolution Image	32
15	Schematic Representation of Position-Dependent Sampling, Which Shows the Concomitant Space/Frequency Trade-off	33
16	A Possible Hardware Implementation of the Gabor Approach, Which Uses Gaze-Slaved, Computer-Generated Imagery	37

EFFICIENT IMAGE GENERATION USING LOCALIZED FREQUENCY COMPONENTS MATCHED TO HUMAN VISION

I. GENERAL INTRODUCTION

The major problem confronting designers of high-fidelity visual simulators is the dual requirement of high resolution and wide field of view. At even moderate light levels the spatial resolution of the human visual system is better than 0.5 minute of arc, and its effective field of view subtends more than 10^4 square degrees. To generate (and update at 60 Hz) an image over this field of view, with sufficient detail to provide full resolution regardless of the operator's point of gaze, would require that visual data be manipulated at a rate exceeding 10^9 bits per second. Obviously, even the most powerful computers cannot perform such a task in a real-time environment; thus, in practice, either resolution or field of view must be compromised (Schachter, 1983).

Conventional computer image generation techniques are inherently inefficient for at least two reasons. First, in order to generate a realistic approximation of a natural (i.e., fully textured) image, conventional techniques require the specification of each of millions of display picture elements (pixels). Second, they allocate image information uniformly across the visual field while the human visual system is distinctly nonuniform in its ability to acquire and process that information. The purpose of the present report is to describe a technique for visual image generation which addresses these two limitations. The proposed technique retains many of the best features of existing image generation techniques and in addition incorporates features designed to generate and present imagery more efficiently.

The computer image generation technique proposed here draws upon such diverse fields as communication theory, visual anatomy, neurophysiology, and imaging technology. We will, therefore, begin with an overview of various terms and concepts which will be used throughout the report. This will be followed by a detailed description of the proposed image generation technique, as well as associated technologies required for its efficient implementation.

II. PERIODIC FUNCTIONS AND FOURIER THEORY

Introduction

Traditionally, there have been two general approaches to image representation. The first involves a point-by-point or pixel-by-pixel (where the term pixel is short for "picture element") specification in the spatial domain. This type of representation is appropriate for describing the spatial operation performed by the first layer of the retina, where an array of approximately 120 million photoreceptors (mostly rods) samples the image. Such a spatial pointwise specification of an image is most suitable for images made up of information confined to small, discrete areas of the visual field. A good example of this would be the image of Figure 1a, which depicts the pattern of stars representing a well-known constellation. For this type of image, the pointwise representation is very efficient in that a small set of numbers, specifying each star's position and intensity, can fully describe the image for its storage, transmission or any other application. If, however, the information is widely distributed over the image, then a large set of numbers is required for specifying its content using the point-by-point representation. Consider for example the image presented in Figure 1b. In this case, the brightness of practically all pixels has to be specified in order to represent the image using a pointwise representation. The repetitive (periodic) structure of the image suggests, however, that there may be a more efficient way of representing the image.

This brings us to the alternative approach for representing an image--namely, the use of periodic components, each of which extends over the entire image and which when added together will represent the image as a whole. An image like that of Figure 1b, for instance, can be generated by adding together only 48 relatively simple luminance distributions. The entire image can therefore be represented by as few as 96 numbers (i.e., the spatial frequency and phase of each of the 48 components) as compared to specifying thousands of individual pixel values. It is desirable in this context to choose a set of components whose properties are such that they can be used to specify (i.e., synthesize or analyze) *any* image. That such a set of components exists was first shown by the famous French mathematician and physicist Jean Baptiste Joseph Fourier. Fourier's technique will be described in some detail below, following the introduction of several basic terms and concepts which will be required here and in later sections.

One-Dimensional Periodic Functions: The Sinewave Grating

Some of the simplest images encountered in image analysis and synthesis (Ginsburg, 1978; Papoulis, 1968) as well as in vision research (Braddick, Campbell, & Atkinson, 1978; Campbell & Maffei, 1974; Ginsburg, 1978) are those whose intensity varies periodically in one dimension only. The intensity distribution of such a periodic image is shown in Figure 2. An example of an image, whose intensity variation is given in Figure 2, is shown in Figure 3a, where it is assumed that the image, being viewed through a circular window, extends to infinity in all directions. The image of Figure 3a contains no structure in the vertical dimension--that is, a

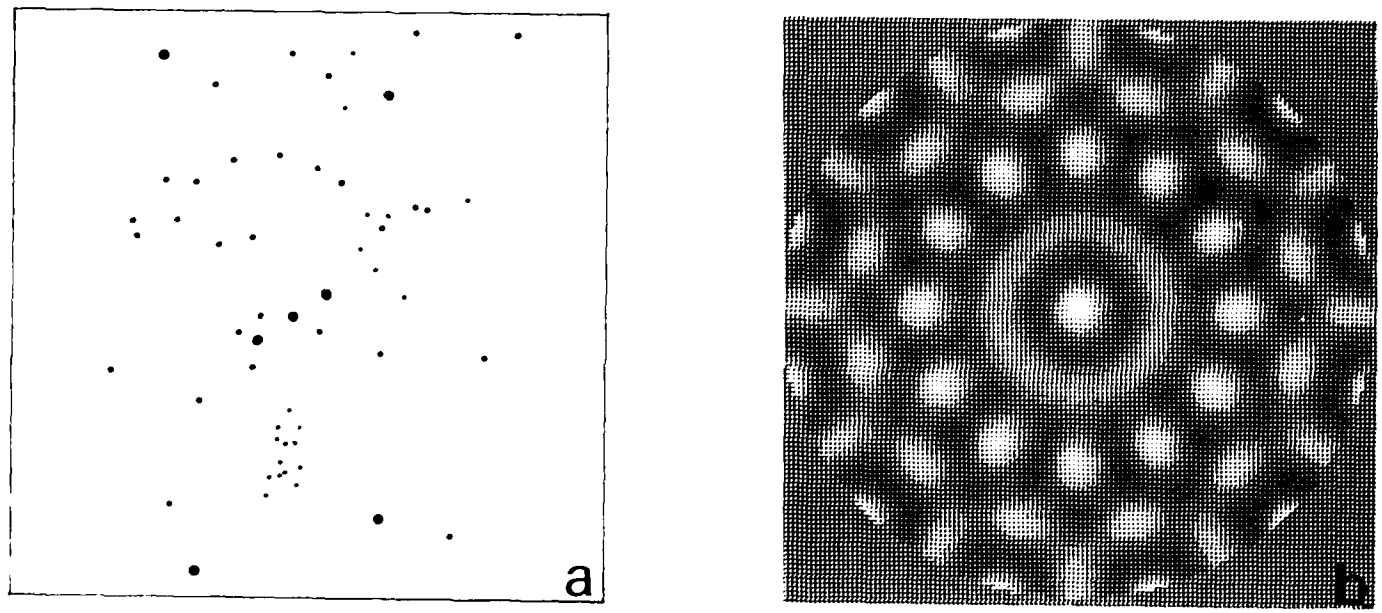


Figure 1. Example of Random and Periodic Images. The images shown lend themselves optimally to either a point-by-point, spatial representation (a), or a spatial frequency representation (b). The image in (a) is a star field representing the constellation Orion, while the image in (b) is a simple texture pattern obtained by adding together 48 sinusoidal luminance distributions.

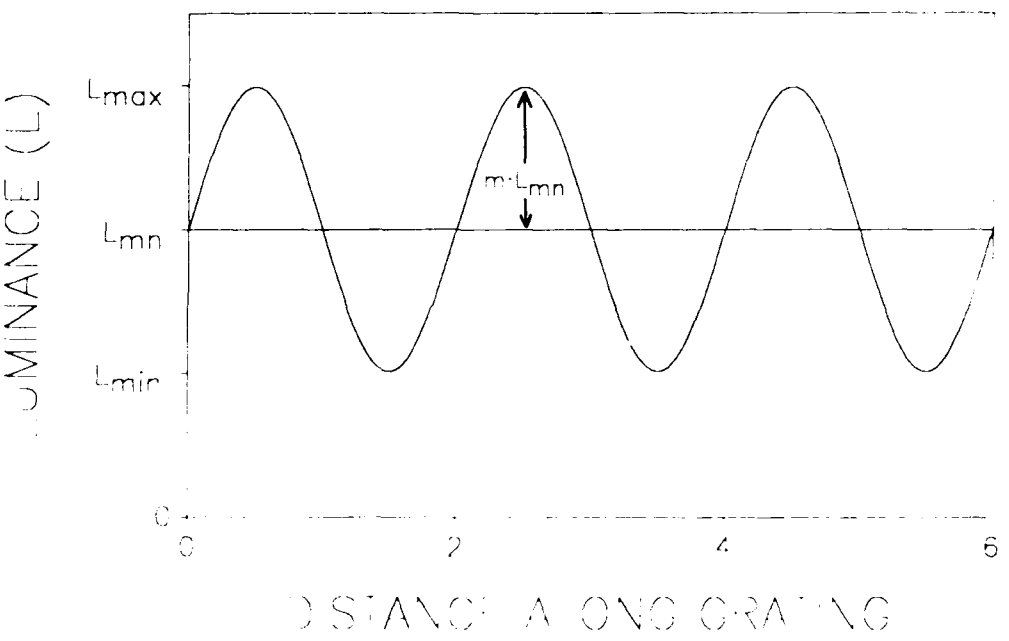


Figure 2. Parameters Which Characterize a Sinusoidal Grating. The plot represents the spatial luminance distribution of a (windowed) cosine wave grating. As can be seen from the figure: (1) $L_{mn} = (L_{max} + L_{min})/2$ and (2) $m \cdot L_{mn} = (L_{max} - L_{min})/2$. Substituting (1) into (2) gives $m = (L_{max} - L_{min})/(L_{max} + L_{min})$, which shows that the modulation is equal to the so-called Michelson contrast.

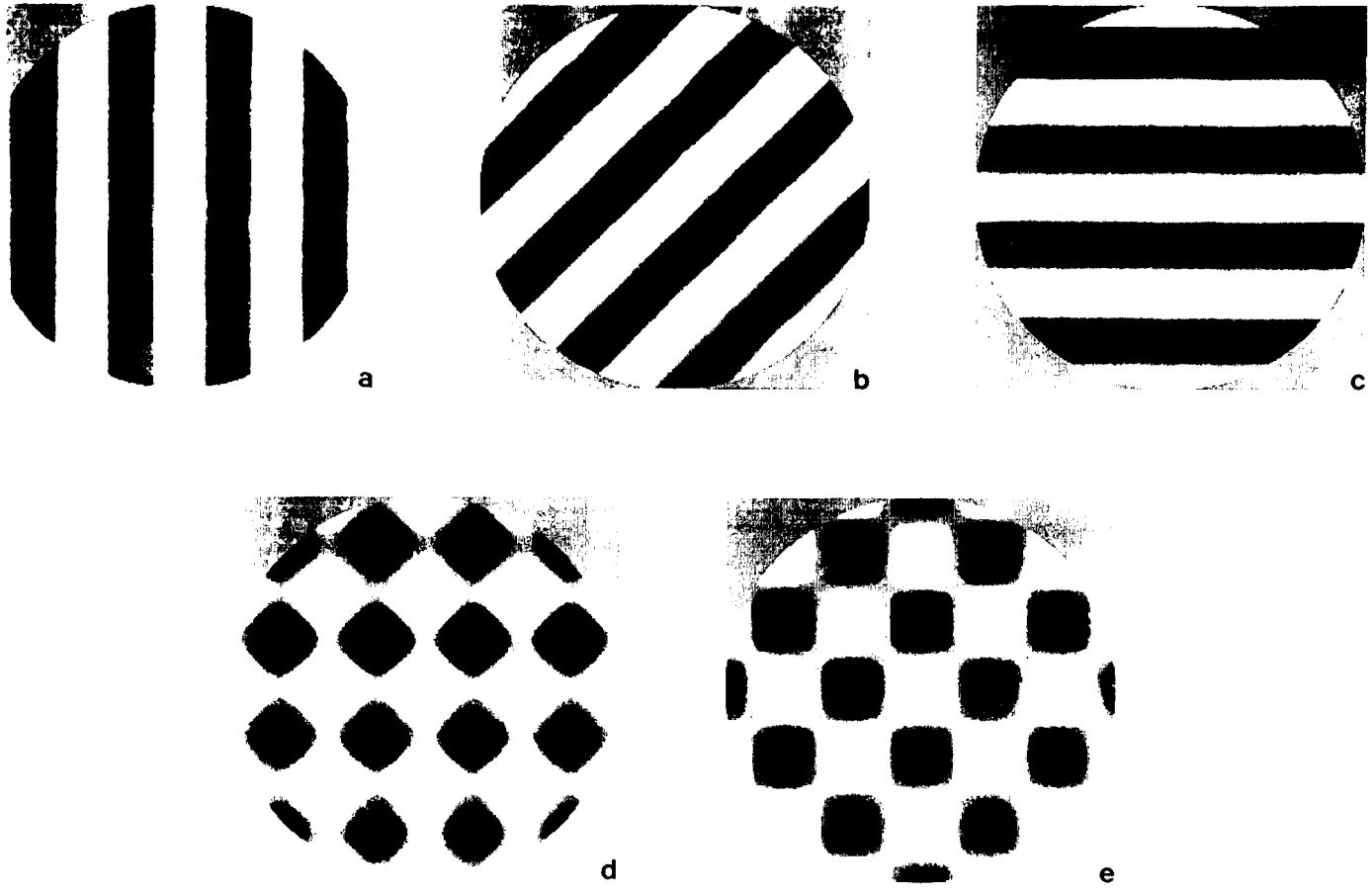


Figure 3. Examples of One- and Two-Dimensional Sinusoidal Gratings. (a), (b), and (c) are single gratings oriented at 90 (vertical), 45, and 0 (horizontal) degrees, respectively. All are one-component gratings although (b) has spatial structure in both the horizontal and vertical dimensions. (d) and (e) are composed of two perpendicularly oriented components and thus exhibit spatial structure in all directions.

constant value would be obtained if image intensity were measured along any vertical path. Thus, although the image is spatially two-dimensional, it can be adequately represented by the one-dimensional function shown in Figure 2. This one-dimensional function is known as a **cosine wave** (abbreviated "cos") and is completely specified by three parameters: its **amplitude (A)**, which is a measure of half the vertical distance between adjacent peaks and troughs; its **mean luminance (L_{mn})**, which is the level about which the sine wave varies; and its **spatial frequency** ($\omega = 2\pi/d$), which is a measure of the number of cycles or peak-to-trough pairs that occur within a given horizontal distance, d. Thus, luminance distributions like those shown in Figure 2 can be fully described by the equation:

$$Luminance(x) = L_{mn} + A\ cos(\omega_x x).$$

In the fields of optics and image science, the amplitude of a cosine luminance distribution is often specified by the quantity $m \cdot L_{mn}$ where m [$= (L_{max} - L_{min})/(L_{max} + L_{min})$, see Figure 2] is known as the **modulation** or contrast of the luminance distribution. By this definition, m varies from 0 (i.e., a homogeneous field) to 1 (i.e., a grating with peak-to-trough amplitude equal to $2 \cdot L_{mn}$). Unless otherwise noted, the following discussion will assume, for simplicity, that L_{mn} for each cosine wave image is equal to its amplitude ($m \cdot L_{mn}$), which is equivalent to the assumption that m is maximal (i.e., equal to 1). The consequence of this assumption is that the minimal luminance, which occurs at each trough of the cosine wave, will be zero rather than some positive number.

The cosine wave described above is defined relative to a reference point about which it is symmetric. This means that the ordinate values of the function are the same for abscissa values equidistant from the origin in each direction. If the cosine wave is translated a distance equal to one-quarter of the distance between peaks, the result is an antisymmetrical function which is called a **sine wave** (abbreviated "sin"). For an antisymmetrical function, the ordinate values corresponding to points equidistant from the origin to the left and right are equally different in magnitude (luminance) from the mean level but are in opposite directions relative to it. [Sine and cosine waves are often collectively referred to as **sinusoids**.] We may conclude from this example that in addition to amplitude, spatial frequency, and mean luminance, the shift along the spatial coordinate relative to the reference point must also be specified in order to fully define a sinusoidal function. The shift is called **phase**, and it is measured in degrees (0-360 degrees) or radians ($0-2\pi$). Clearly, any addition of multiples of 2π (or 360 degrees) will not affect the relative position of the function. When phase, ϕ, is taken into account and remembering that $A = m \cdot L_{mn}$, the equation describing the sinusoidal grating becomes:

$$Luminance(x) = L_{mn} + m \cdot L_{mn}\ cos(\omega_x x + \phi)]$$

$$= L_{mn}\ [1 + m \cdot cos(\omega_x x + \phi)].$$

It should be noted that a sinusoid of any phase can be obtained simply by adding together one sine and one cosine function of the same spatial frequency, providing that their amplitudes can be varied appropriately. This is a consequence of the trigonometric identity $\sin(a+b) = \sin(b)\cos(a) + \cos(b)\sin(a)$. If the quantity b on the left side of the equation is interpreted as a phase shift, then the terms sin(b) and cos(b) on the right side are constants representing the amplitudes of the sinusoids [namely, cos(a) and sin(a)] with which they are associated. Clearly,

then, the sinusoid of arbitrary phase represented by sin(a+b) can be obtained by adding together two sinusoids of the same spatial frequency if the amplitudes of the latter can be varied as required. Although this is a simple and well-known relation, its practical consequences are not often noted. We will return to this point in our discussion of the frequency representation of sine wave gratings.

Two-Dimensional Periodic Functions

Spatial structure in two dimensions can be introduced by changing the orientation of the sinusoid of Figure 3a, thus producing the image shown in Figure 3b. The image of Figure 3b is similar to that of Figure 3a except that it has been rotated 45 degrees in the clockwise direction. The effect of a change in orientation can be seen in the intensity representation of this image. The amplitude (or contrast) of the function has not changed but its horizontal frequency has. In addition, the rotated image now exhibits spatial structure in the vertical dimension. Thus, the functional form of the image now contains both horizontal ($\omega_x x$) and vertical ($\omega_y y$) spatial frequency terms and may be expressed as:

$$Luminance\ (x,y) = L_{mn}\ [1 + m \cdot cos(\omega_x x + \omega_y y).$$

As is obvious from this expression, a vertical grating (Figure 3a) is the special case of an oriented grating (Figure 3b) whose spatial frequency in the vertical direction (ω_y) is zero. Similarly, a horizontal grating (Figure 3c) has a non-zero vertical spatial frequency but a horizontal spatial frequency (ω_x) of zero.

Consider next the image shown in Figure 3d which shows two sinusoids added at right angles to each other, resulting in a multicomponent two-dimensional grating. If the intensity of this image were measured along any horizontal (or vertical) path, a function of the same general form as that shown in Figure 2 would result. However, the mean luminance of these functions is now dependent also on the intensity variations in the complementary orientation. This dependence of image intensity, on both the vertical (Figure 3a) and horizontal (Figure 3c) components making up the image, pertains irrespective of the path along which it is measured. The equation representing the grating shown in Figure 3d is:

$$Luminance(x,y) = L_{mn}\ [1 + m \cdot cos(\omega_x x) + m \cdot cos(\omega_y y)]$$

Finally, rotating the multicomponent image of Figure 3d by 45 degrees results in the image shown in Figure 3e and the following functional representation:

$$Luminance(x,y) = L_{mn}\ [1 + m \cdot cos(\omega_x x + \omega_y y) + m \cdot cos(\omega_x x + \omega_y y)]$$

Note that when two luminance distributions with the same mean luminance are added together, the mean luminance is doubled. In order to maintain the same mean luminance in the multicomponent image as in its components, it is necessary to halve the mean luminances of the components before adding them. In the resulting scaled image, the majority of the horizontal and vertical luminance crosscuts have a mean luminance different from that of the entire image and hence of the individual components of the image. Further, none of the horizontal or vertical crosscuts of the multicomponent image shown in Figure 3d exhibit the full peak-to-trough luminance of the components. However, along the major diagonals of

the image (at 45 and 135 degrees) and for periodically spaced crosscuts parallel to them, both the mean luminance and the peak-to-trough luminance are the same as in the original component images. The spatial frequency of the periodic structure along the major diagonals is lower, by a factor of the square root of two, than that of either of the component images. Thus it is evident from the images shown in Figures 3d and 3e that complex luminance variations can emerge when as few as two simple luminance distributions are combined. As will be demonstrated later, the complexity of multicomponent images further increases when the spatial frequency and phase of the individual components are varied.

Frequency Representation of Simple Images

As noted earlier, sinusoidal images can be fully described by their amplitude, mean luminance, spatial frequency, and phase. Because only three numbers (recall that we are assuming that amplitude = mean luminance so that m = 1) are required to specify an image like that shown in Figure 3a, the luminance distribution across such an image, technically composed of infinitely many points, may be considered excessively complex. An alternate method for conveying the information contained in Figure 3a is shown in Figure 4a, where the horizontal axis now represents spatial frequency in units of cycles per millimeter and the vertical axis represents amplitude. Figure 4a may be described as a representation in **one-dimensional (1-D) spatial frequency space**. As is evident from the figure, two functions (each shown as an arrow representing an amplitude and a spatial frequency) in this space are sufficient to describe any image of the type represented by Figure 3a.

The functions represented by the arrows in Figure 4 are known as **Dirac delta-functions (δ-functions)**. These functions are assumed to have zero width and infinite height so that, though the function technically exists at only one point, the area under the function is equal to 1. Because it is difficult to draw a function of infinite height, δ-functions are by convention represented by an arrow whose length corresponds to the area under the function. When δ-functions are used in the context of grating images, the height of the δ-function is related to the magnitude (contrast) of the grating, and its distance from the origin is related to the spatial frequency of the grating.

[Note that an infinite homogeneous field would be represented in frequency space by a single δ-function located at the origin (i.e., corresponding to a spatial frequency of zero). Recall also that all grating images are, in effect, sinusoids added to a homogeneous field. Therefore, all frequency representations of these images should include a component at the origin. For the sake of simplicity, however, we have chosen not to include this component in our figures.]

Although one δ-function in the 1-D space of Figure 4a would suffice to specify spatial frequency and amplitude, a second point is required to specify the phase of the sinusoid. This concept is illustrated in Figures 5a-d which depict, respectively, the 1-D spatial frequency representations for the functions $Y = L_{mn} \sin(\omega x)$, $Y = L_{mn} \sin(\omega x) + 30°$, $Y = L_{mn} \sin(\omega x) + 60°$, and $Y = L_{mn} \sin(\omega x) + 90°) = \cos(\omega x)$. The upper and lower diagrams on the right are the conventional representations for the sine and cosine functions, respectively, wherein each δ-function is of unit length. Differences in phase are represented by different relative lengths

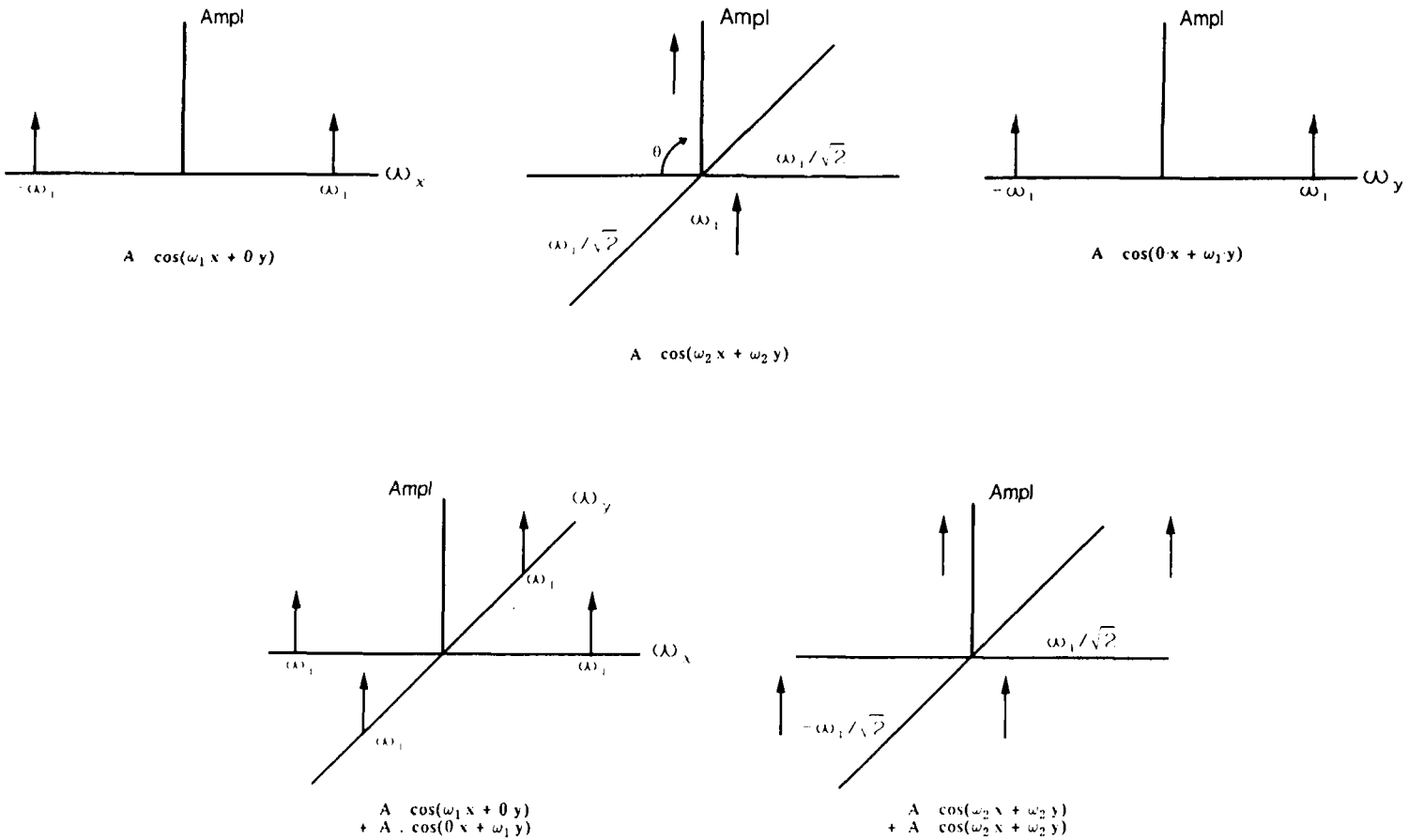
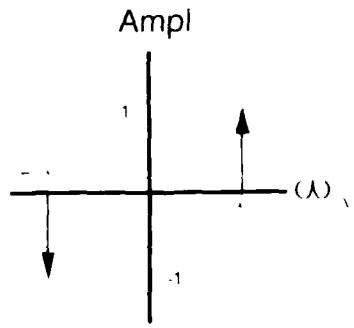
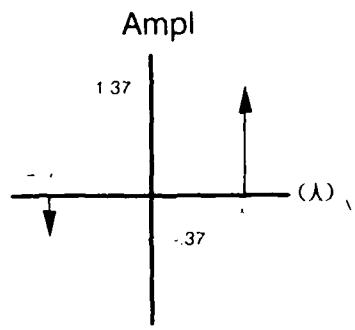


Figure 4. Frequency Representation of the Gratings Shown in Figure 3.

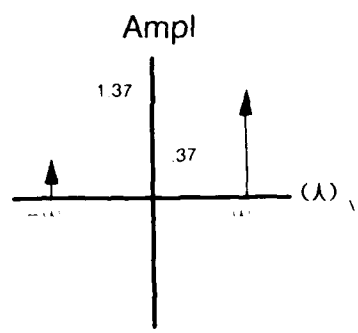
$$\sin(\omega x + 0) = (\cos 0)\ \sin \omega x + (\sin 0)\ \cos \omega x$$
$$= 1\ \sin \omega x + 0\ \cos \omega x$$
$$= \sin \omega x$$



$$\sin(\omega x + 30) = (\cos 30)\ \sin \omega x + (\sin 30)\ \cos \omega x$$
$$= 0.87\ \sin \omega x + 0.50\ \cos \omega x$$



$$\sin(\omega x + 60) = (\cos 60)\ \sin \omega x + (\sin 60)\ \cos \omega x$$
$$= 0.50\ \sin \omega x + 0.87\ \cos \omega x$$



$$\sin(\omega x + 90) = (\cos 90)\ \sin \omega x + (\sin 90)\ \cos \omega x$$
$$= 0\ \sin \omega x + 1\ \cos \omega x$$
$$= \cos \omega x$$

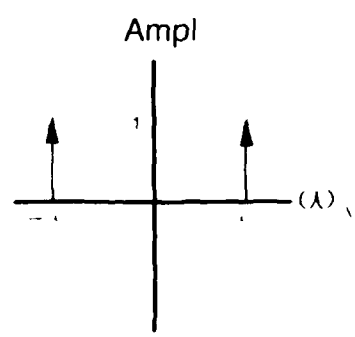


Figure 5. Phase Representation in Frequency Space. Calculations and diagrams showing how a sinusoid with various phases can be represented by appropriately weighted sine and cosine functions.

of the two δ-functions representing the grating. The procedure for calculating those relative lengths is shown to the right of each diagram and consists simply of combining the sine and cosine representations with different relative weights.

The images of Figures 3b, 3d, and 3e also can be represented in the spatial frequency domain, as shown by the corresponding plots in Figure 4. Because these images are two-dimensional, a second spatial frequency axis is required for representation in what may be called **two-dimensional (2-D) spatial frequency space**. In this space, the image of Figure 3b is represented by two δ-functions located along the same line in the ω_x-ω_y plane (see Figure 4b). This line is not collinear with either of the spatial frequency axes and so the image represented by the two δ-functions on that line can be projected onto both of the orthogonal axes--that is to say, it is an oriented grating. The magnitudes of the projections onto the two axes are ω_1 and ω_2, and so the spatial frequency of the grating measured along an axis orthogonal (i.e., at 90 degrees) to its orientation is equal to $\sqrt{\omega_1^2 + \omega_2^2}$. The orientation of the grating making up the image of Figure 3b is now represented by the angle labelled θ, which in this case is 45 degrees and can in general be obtained by the formula $\tan \theta = \sin \theta / \cos \theta = \omega_x / \omega_y$.

The image of Figure 3d, although it is also two-dimensional, is different from that of Figure 3b in that it is composed of two gratings at right angles to each other. The image of Figure 3d may be represented in the 2-D spatial frequency space by two pairs of δ-functions, with one pair located along each of the orthogonal frequency axes (see Figure 4d). The phases of the two gratings are equal and so the δ-functions in each pair have the same amplitude. Also, because the two component gratings have the same spatial frequency, the δ-functions are equidistant from the origin. Finally, the image of Figure 3e may be represented in 2-D spatial frequency space by the four δ-functions shown in Figure 4e, which are no longer on the ω_x and ω_y axes but which are the same distance from the origin as the δ-functions of Figure 4d. As was the case for the single grating shown in Figure 3b, the spatial frequencies and orientations of the two gratings of Figure 3e are represented by the projections of each pair of points on the two axes.

The three-dimensional space shown in Figures 4b, 4d, and 4e is difficult to depict; so in situations where the spatial frequency of image components is of primary importance (and the amplitude is either constant or can be specified separately), two-dimensional spatial frequency information is often represented as shown in Figure 6. Here the horizontal and vertical axes represent spatial frequencies along the two spatial dimensions of the image. In this representation, any point which falls on either of the axes corresponds to a one-dimensional grating--a vertical grating if it falls on the ω_x axis and a horizontal grating if it falls on the ω_y axis. Any other point will have projections along both axes and hence will represent a one-component, two-dimensional (oriented) grating. Consider, for example, the point P shown in Figure 6. The grating represented by this point has projections whose magnitudes are ω_1 and ω_2, as was discussed earlier in reference to Figure 4b.

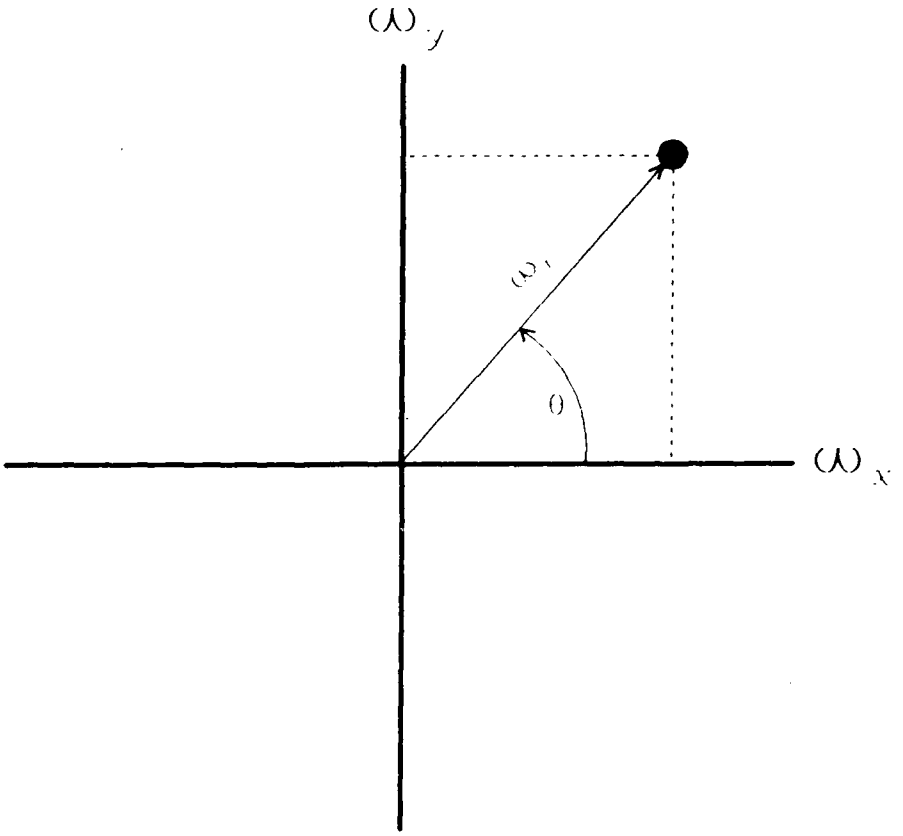


Figure 6. Two-Dimensional Spatial-Frequency Space. The simplified two-dimensional space includes both of the spatial-frequency axes shown in Figures 4b, 4d, and 4e.

Spatial Frequency Bandwidth

The spatial frequency representations shown in Figure 4 apply only to sinusoidal images of infinite extent. Only in the case of an infinite grating is it appropriate to represent the luminance distribution by a single spectral function of infinitesimal width (i.e., a δ-function). Obviously, real images are limited in their spatial extent and as such may be considered to be a product of an infinite image and a finite window. An example of the window function alone is shown in Figure 7a. This function is simply a homogeneous field of limited extent whose luminance is equal to the mean luminance of the gratings shown in Figure 7b (and other figures presented in this report). As was discussed earlier, an infinite homogeneous field can be represented by a single spectral component (δ-function) which is positioned at zero frequency and whose amplitude is related to the field luminance. However, if a homogeneous field is restricted in its *spatial* extent, the resulting associated spectral distribution becomes continuous and theoretically infinite in its *spectral* extent. The spectral distribution associated with the window function of Figure 7a is known as a **sinc-function** and is shown in Figure 7b. This distribution, which peaks at zero spatial frequency, displays multiple lobes whose peak magnitude progressively decreases with distance from the origin.

The grating shown in Figure 7c, like those shown in previous figures, represents the product of an infinite sinusoidal grating and the window function of Figure 7a. The frequency representation of Figure 7c will therefore reflect the contribution of both the infinite grating and the window. Because the grating shown in Figure 7c is generated by the product of two spatial functions, its frequency representation can be obtained by combining the spectral distributions of the two spatial functions by an operation known as **convolution**. We will not discuss convolution here (see, e.g., Bracewell, 1986, for details) except to note that in the case where one of the two spectral distributions is a δ-function, the resulting combined distribution is obtained by shifting the other spectral distribution to the position of the δ-function. So, for instance, given a fixed window function (representing, say, a visual display), changing the frequency of a grating presented within this window will result in identical spectral distributions (determined by the window) but located at different points along the frequency axis.

To gain some insight into the effects of the window itself on the resulting frequency representation, consider the 1-D, windowed sinusoids shown in Figure 8. Each sinusoid has the same spatial frequency but a different spatial extent. A graphical representation of the luminance distribution corresponding to each sinusoid is shown as the uppermost of the two graphs located to the right of each image. The lowermost member of each pair of graphs shows the sinc-function corresponding to the associated grating image. Changing the width of the window does not affect the position of the sinc-functions along the frequency axis. However, as the window decreases in width, there is a concomitant increase in the width of the lobes of the sinc-function. The result is that component energy is redistributed such that relatively more of that energy is associated with frequency components farther from the single frequency component contributed by the infinite grating.

Figure 7. A Grating Image as the Sum of an Infinite Grating and a Window Function. (a) A homogeneous field representing the window function which is superimposed, in effect, on all finite images. (b) A finite grating obtained by combining the window function of (a) with an infinite grating. The predominant spatial frequency of the images in (a) and (b) are different, but the window function has not changed and so neither has the bandwidth of the image spectrum.

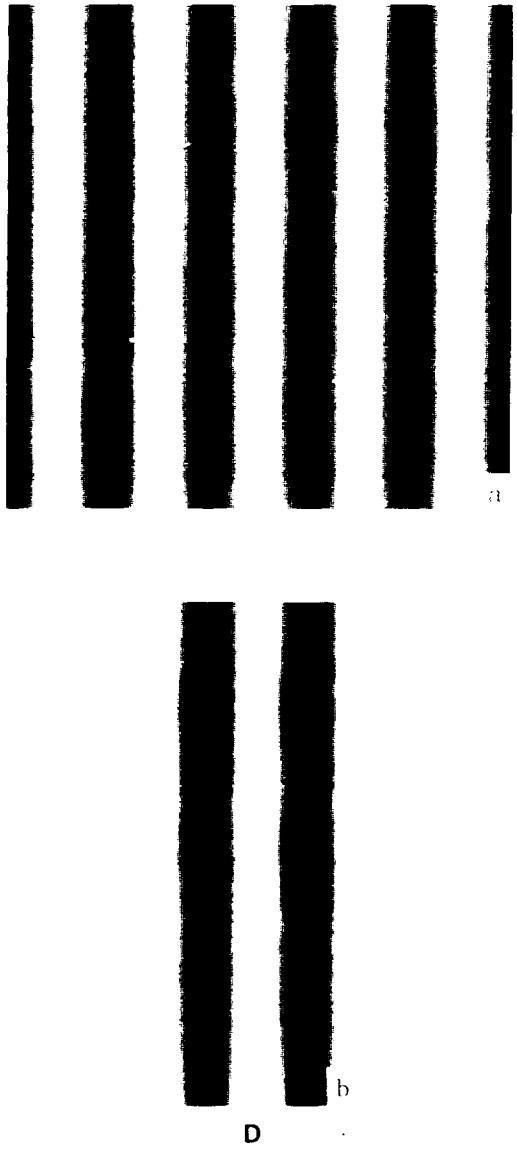


Figure 8. Dependence of Image Bandwidth on the Spatial Extent of the Window Function. (a) A finite grating of spatial frequency ω and its frequency representation. (b) The same grating as in (a) except combined with a narrower window function. Reducing the spatial extent (width) of the grating increases its effective spatial frequency bandwidth.

The range of spatial frequencies associated with spectra like those shown in Figures 7 and 8 is often quantified by defining an effective width of the central lobe. This effective width, which is usually determined for an amplitude which corresponds to one-half of the peak amplitude, is called the **spatial frequency bandwidth**. With the exception of those associated with images of infinite spatial extent (which are obviously unrealizable in practice), *all* frequency spectra contain *all* spatial frequencies. However, in practice only those components whose magnitude (or energy) is above a certain value are of interest, and so the spectrum is considered to have a finite extent. Thus, the spatial frequency bandwidth is not a measure of the number of frequency components making up a sinusoidal image but rather, of the relative magnitude of the frequency components which are close to the frequency of the sinusoid itself. In the context of visual information processing, for instance, the advantage of a narrow bandwidth image is that there will be fewer frequency components present which might interfere with detection of the signal of interest (the sinusoid in this case).

Basis Properties, Orthogonality, and Other Characteristics of Sinusoids Which Make Them Useful for Representing Images

We will discuss in the next section the advantages inherent in producing images by adding together simple components. To be sufficiently general, a set of such components must *span* the space of all required images, which means that it can be used to generate any image which belongs to the defined space. A set which meets this requirement is called a *basis*. In the most general case, it is required that the set of functions which constitutes a basis can generate any possible image. It is usually convenient that the set of coefficients, which determines how much of each component must be used to produce the required image, be unique and easily determined. For this to be the case, all of the components that constitute the basis must be *orthogonal*. The concept of orthogonality is best discussed in the mathematical context of inner products (Strang, 1986), which would be inappropriate here. In the present context, orthogonality is roughly synonymous with independence in that it implies that no component can be obtained by adding together any of the other components. The sine and cosine functions mentioned earlier provide an example of an orthogonal basis. The simple trigonometric identity noted earlier demonstrated that, for a given spatial frequency, *any* sinusoid could be produced by adding together appropriately weighted sine and cosine functions. Thus, we may say that the set of sine and cosine functions (the basis set) spans the space of the given sinusoid over all possible translations (i.e., phases).

As mentioned above, the advantage in using an orthogonal basis is that the resulting coefficients, which are the weights associated with the functions constituting the basis, are unique and are relatively easily determined. However, a given function (or image in the present context) can also be represented by nonorthogonal bases. Considering again the example of the translated sinusoid, the components in that case were sine and cosine functions which are orthogonal in the sense that one is phase-shifted by 90 degrees relative to the other i.e., $\sin(\omega x) = \cos(\omega x - 90^\circ)$. However, a basis could be formed in this case by using pairs of sinusoids with other phase relationships. For instance, the component $\cos(\omega x)$ could be replaced by $\sin(\omega x - \phi)$, where ϕ is any desired phase. The lack of orthogonality of the components in the latter

case means that both a sine and a cosine term are now required to express the function which replaced the original cosine term. Thus, the difficulty in working with nonorthogonal bases is that the coefficients cannot be determined independently. We will return to this point in a later section.

We have noted that sine and cosine functions are suited to image analysis and synthesis because they form a basis set which is orthogonal. However, many other functions also form orthogonal bases (Higgins, 1977); so, this fact alone does not explain the popularity of sinusoids for this purpose. We have also noted that the sine and cosine functions are particularly efficient and easy to deal with computationally. However, it may be argued that this is not so great an advantage given that powerful computers are now so readily available. There are also other reasons for the popularity of sinusoids in image analysis. First, sinusoids are an integral part of linear systems theory, which is the most powerful theory in the field of signal analysis. Specifically, this theory uses the concept of transfer functions whereby the response of a system, to each frequency component of interest, is specified. Once the transfer function is obtained, the response of the system to any signal (an image in the present context) can be predicted by simply multiplying the frequency representation (i.e., spectrum) of the signal by the transfer function and then computing the inverse transform. Second, many natural phenomena have resonant properties and, when they are finely tuned and do not dissipate energy, they often exhibit sinusoidal behavior. Further, because more complex natural phenomena result from the combined activity of a number of simpler resonating subsystems, they can quite often be characterized by combining a number of sinusoids which is small relative to the number of functions required using any other universal basis. Various studies with natural (textured) images indicate that important attributes of images are periodic and can be well defined by sinusoids.

Frequency Analysis and Synthesis of Complex Images (Fourier Theory)

We will now briefly describe a well-known approach to image representation using orthogonal components. A fundamental theorem, credited to the mathematician Jean Baptiste Fourier (and extended to two-dimensional functions), states that the set of two-dimensional sine and cosine functions spans the space of real-valued two-dimensional functions. In other words, any image, considered to represent one cycle of a two-dimensional periodic image, can be generated from a weighted linear sum of the set of sine and cosine functions:

$$s(x,y) = \sum_{n_x n_y}^{\infty} [a_{n_x n_y} \cdot \sin(n_x \omega_x x + n_y \omega_y y) + b_{n_x n_y} \cdot \cos(n_x \omega_x x + n_y \omega_y y)]$$

In this equation, the function s(x,y) represents any two-dimensional image and the pairs of sinusoids, sin(_) + cos(_), are the components forming the basis. Each component pair corresponds to a particular spatial frequency (ω), and the sets a.. and b.. are the coefficients, or weights, of the sine and cosine elements associated with the given frequency component. It is these sets of coefficients which must be determined to uniquely represent an image in terms of its frequency components. That is, once the coefficient sets are determined, the image can be

reconstructed by adding together the appropriately weighted pairs of sinusoids, as indicated by the summation symbol, Σ. The only difference between the components referred to in the equation shown above and the components in the simple trigonometric example presented earlier is that the former are composed of sine-cosine pairs, are two-dimensional, and are more numerous. Although we will not describe here the mathematical details of the transform which determines the coefficients, we reiterate that those transform techniques (see e.g., Bracewell, 1986) are relatively simple and efficient because the chosen basis is orthogonal.

Relevance of Fourier Analysis to Vision

The idea of using visual stimuli composed of frequency components was first presented by the physicist Ernst Mach, who, in 1866, designed a mechanical device for adding frequency components of arbitrary amplitude and phase, and whose general contributions to the field of vision are very well known (see Ratliff, 1972). Schade (1956), who was concerned with neural processing in the early stages of the visual "luminance channel," applied the spatial frequency approach in his construction of a photoelectric analog of the visual system. Although it was clear that images could be fully represented by spatial frequency components, and that linear (and spatially uniform) systems could be characterized by their response to those frequency components, the question remained as to whether the frequency decomposition approach was relevant to visual (i.e., neurophysiological) processing. As it turns out, responses to simple visual stimuli in the form of sine wave gratings and to complex stimuli composed of the sum of as few as two sine wave gratings of different frequencies can elucidate fundamental properties of the visual system (Campbell & Maffei, 1974). Campbell and Robson (1968) showed further that the human visual system has the highest sensitivity to contrast for spatial frequencies near three cycles per degree, with sensitivity dropping off at higher and lower spatial frequencies. This contrast sensitivity function is taken to represent the so-called **modulation transfer function (MTF)** of the visual system.

Campbell and Robson (1968) also investigated whether the visual system breaks down an image projected onto the retina into spatial frequency bands in a manner analogous to the decomposition of auditory signals by the ear. Given a sine wave grating of frequency ω_0 (the **fundamental**), we can define an **harmonic** to be any grating whose spatial frequency is an integer multiple of ω_0. If we decompose a square-wave grating in accordance with Fourier's theory, we find that it is composed of the sum of the odd harmonics, each having a contrast inversely proportional to the harmonic number. Because the fifth and higher harmonics have very low contrast (compared to the fundamental), and because visual sensitivity to these high frequencies is relatively low, if the fundamental grating is added to a third harmonic (i.e., a grating whose spatial frequency is $3\omega_0$) and if the contrast of the third harmonic is one-third that of the fundamental, then the combination may be expected to resemble a square wave if indeed the visual system performs a Fourier-like analysis. This is exactly what Campbell and Robson found, thus supporting this conclusion at least for certain simple visual stimuli.

The rationale of the Campbell and Robson experiment described above was as follows: If there exists a visual mechanism which is selectively tuned to a relatively narrow band around the spatial frequency of the third harmonic, then it should be possible to reduce the mechanism's sensitivity by adapting the visual system to a high-contrast version of that harmonic. And, further, if the visual system processes and transmits the first and third harmonic by different mechanisms, and if the mechanism responsible for processing the third harmonic can be "turned off" or alternatively if its sensitivity (or gain) can be reduced, then a square wave should appear to an observer to look like a sine wave grating (whose frequency is that of the fundamental or first harmonic). This is exactly what Campbell and Robson found; thus, they concluded that different frequency components (which may differ also in orientation) are transmitted by different channels. This conclusion is consistent with the discovery of Hubel and Wiesel (Hubel, 1982) that there are cells in the visual cortex which respond to bars of a particular width and orientation, and with the findings of DeValois, Albrecht, and Thorell (1982) that such cells respond preferentially to a band of spatial frequencies which is 1-2 octaves in width.

When discussing Fourier analysis in the context of vision, it is important to consider the unique role of component phase. It is well known that the visual system is very sensitive to relative position information (Westheimer, 1978), and when analyzing an image by transform techniques, position information is described by the phase relationship of the frequency components. Indeed, Fourier phase (i.e., the distribution of phase across the entire frequency spectrum which constitutes an image) captures all of the edge information in an image; thus, the amplitudes of the various Fourier components (and hence the image itself) can be reconstructed from the phase information only (Oppenheim & Lim, 1981). To illustrate the importance of phase, and to show the dependence of coherent image structure on the degree of phase specificity, we have generated a sequence of images whose components have the same spatial frequency, orientation, and magnitude, but different phases. The symmetrical and perceptually coherent image in Figure 9a was obtained by phase-locking the components such that one luminance peak of each component coincided with the center of the image (i.e., all components were cosine functions with zero phase). The other images in the series were obtained by progressively increasing the range over which the component phases were randomly distributed. As is apparent, the perceived coherence of the image breaks down when the phase is randomized.

Image Representation Using Nonorthogonal Bases

We noted earlier the advantages in using an orthogonal basis. The practical disadvantage in using a *non*orthogonal basis is that optimization procedures, which involve iterative adjustment and updating of previously estimated components as new ones are computed, are then required to determine the coefficients. These procedures are computationally more intensive than those required to determine the coefficients associated with an orthogonal basis, and the problem is exacerbated as the number of components constituting the basis increases. Nevertheless, it is often advantageous, for other than computational reasons, to use a basis that is not orthogonal.

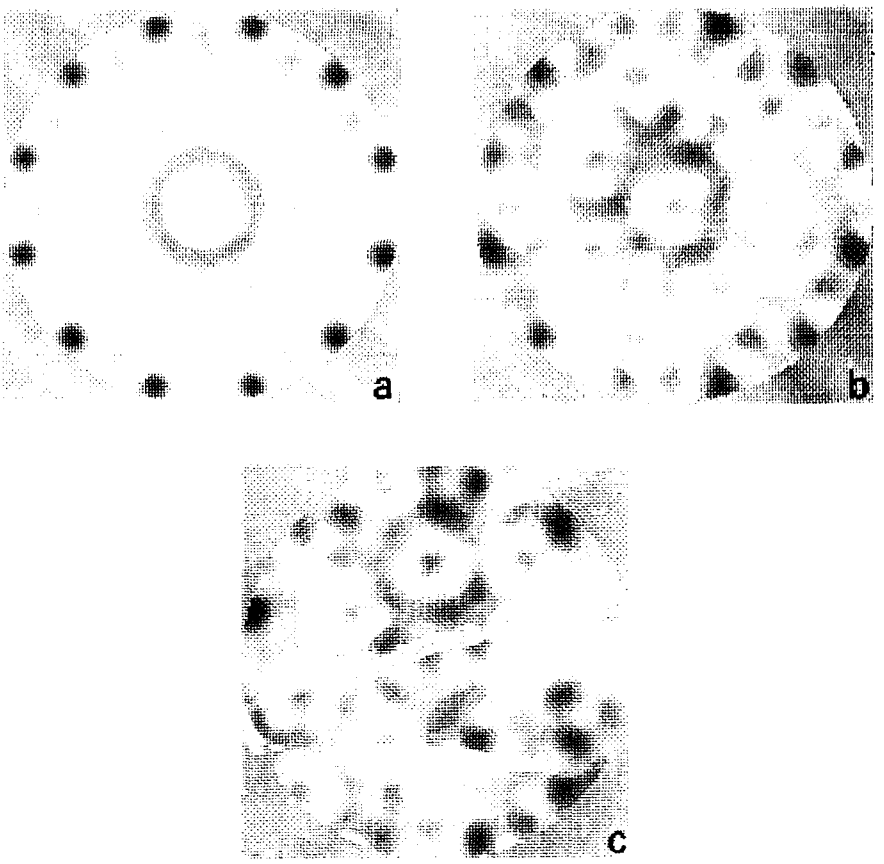


Figure 9. Dependence of Perceived Coherence on Component Phase. (a) An example of a highly structured image which is both periodic and symmetric. The image was generated by adding together only 24 cosine components (4 spatial frequencies at each of 6 orientations). All components were phase-locked and of unit amplitude, and so the entire image can be specified by only 48 numbers. (b) and (c) Images generated by adding together the same 24 components as in (a) except that the component phases have been unlocked and distributed over progressively larger ranges.

In fact, the formalism to be presented in the next section uses a nonorthogonal basis and so we will provide here a qualitative description of the rationale and techniques which will be presented later in greater detail.

Recall that image representation, as we have discussed it, involves first analyzing an image by finding an appropriate coefficient set, and then resynthesizing the image by first multiplying the functions making up the basis set by the associated coefficients and then adding up the resulting products. The set of so-called Gabor elementary functions (GEFs) was chosen as the basis set for the analysis to be presented below because it consists of functions which efficiently match the human visual system. When there are compelling reasons for using a nonorthogonal basis, specialized mathematical techniques can often be used to overcome the concomitant difficulties. For instance, and as is the case for the formalism to be described in the next section, it is possible to find a set of functions, complementary to that originally used in the decomposition (analysis) of the image, and use it in the reconstruction (synthesis). The second set of functions are called auxiliary functions and they must have a one-to-one correspondence with the functions of the nonorthogonal basis which was used in the analysis. The auxiliary functions must be biorthogonal to the original functions in the sense that each of the auxiliary functions must be orthogonal to all of the functions of the original set except for the one which corresponds to it (see, e.g., Higgins, 1977, for more details). Once the second set is determined, the roles of the two sets can be interchanged if desired--that is, either set can be used for the analysis and the other then used for the resynthesis (see Figure 10).

Another approach to dealing with a nonorthogonal basis set is to orthogonalize it using the Gram-Schmidt method. This is a well-known and often-used technique, the details of which may be found elsewhere (Strang, 1986). It should be noted that the orthogonal basis that results from this or related techniques may not share certain of the properties which made the original nonorthogonal basis attractive. For instance, if the set of GEFs were orthogonalized, they might no longer approximate the form of human receptive field profiles.

III. THE GABOR SCHEME: IMAGE REPRESENTATION IN THE COMBINED POSITION-SPATIAL FREQUENCY SPACE

Introduction

The spatial and spatial frequency approaches to image representation are not mutually exclusive. In fact, given that natural images are generally composed of both periodic and discretely localized information, they are most efficiently represented by a scheme which incorporates aspects of both approaches. There are several advantages of this combined approach which may be appreciated by considering local changes in information distribution over the visual field and their effect on the image representation. First, the frequency approach is limited in that a local change, such as the movement of a small object within the image, often requires that the majority or even all of the coefficients (Fourier, Hadamard, etc.) be recomputed. In the combined scheme, however, only those limited number of components which represent the localized object will be affected. Second, object movement is more efficiently coded in

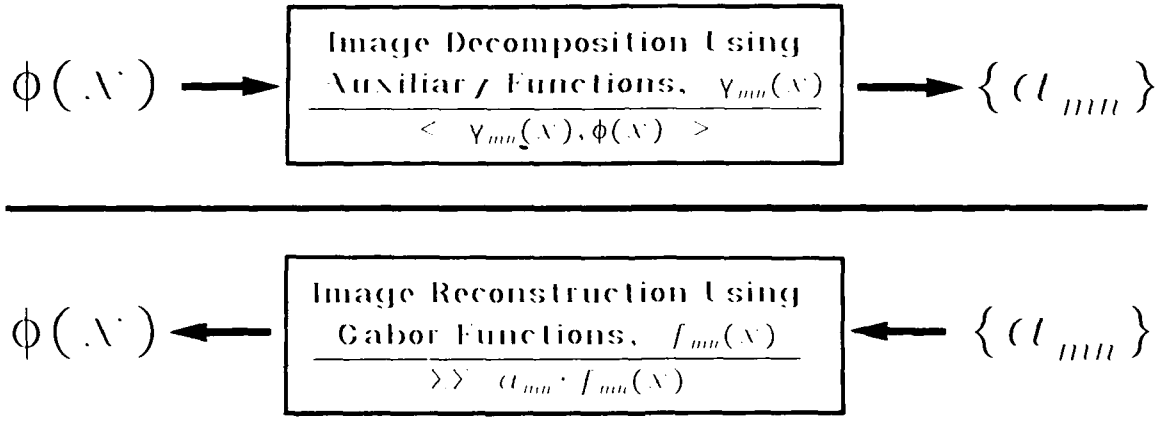


Figure 10. Forward and Inverse Gabor Transforms.

the combined frequency-position space due to the fact that the structure of the object does not change when the object moves across the field, and so only the spatial (and not the spatial frequency) specification of the coefficient distribution needs to be updated. Finally, although the frequency approach is useful for global image processing, it is not appropriate when nonuniform sampling is required, and it is not efficient for representing images which vary in their spectral content from area to area. Thus, a technique is desired wherein local operators are confined to an effective area, which varies as a function of the position in the field, and which can extract frequency signatures in a manner similar to the way this is accomplished by the global operators. In other words, we are looking for a technique of short-distance (analogous to short-duration) frequency analysis which can incorporate variable resolution.

As noted earlier and as will be described in greater detail below, the components of the spatial and spatial-frequency representations are a single point in space and a spectral component, respectively. A singular point in space provides infinite spatial resolution while its frequency representation spans the entire spectrum. Likewise, a spatial-frequency (spectral) component, which is characterized by an infinitely narrow δ-function along the spatial-frequency axis, extends across the entire spatial axis. Thus, the two components are elements of complementary representations, and it would obviously be useful to specify a function which is most narrowly tuned simultaneously in both its spatial extent and its spatial-frequency bandwidth. In the early 1920's, communication engineers attempted to devise such functions and, before succeeding, often tried to transmit a given amount of information per unit time using what was later found to be less than the minimal required frequency bandwidth. As was noted by Gabor (1946), these attempts were analogous to trying to construct a perpetual motion device, in that, analogous to the principle of conservation of energy, there exists a principle which imposes certain constraints on the types of signals that can be physically realized.

Gabor (1946) was concerned with problems related to the efficient transmission of signals, and therefore with the "linkage between uncertainties in the definitions of time and frequency." As further noted by Gabor, these problems were at about the same time beginning to interest researchers in the areas of physics and communication theory. Nyquist (1924), working at Bell Laboratories, proved that the number of telegraph signals which can be transmitted over a communication channel is proportional to that channel's frequency bandwidth. This important observation laid the foundation of modern signal theory. Hartley (1928) generalized this concept by showing that the total amount of information which may be transmitted over such a channel, or the number of degrees of freedom available over the channel in a given time, is proportional to the product of the signal bandwidth and the time available for the transmission. Hartley's paper appeared at about the time that Heisenberg formulated the principle of uncertainty in the context of quantum mechanics. The essence of this principle is that canonically conjugate, observable, physical, quantities like position (along the spatial coordinate) and spatial frequency cannot be simultaneously defined in an exact way (i.e., with infinite resolution). That is, the product of the effective width of a signal in time and the signal's bandwidth can never be less than a value which represents an intrinsic uncertainty. In other words, uncertainties are inherent in the simultaneous definition of position and spatial frequency such that their joint product

must equal or exceed a certain minimal value. Using the formalism of operations with complex exponentials, where each complex exponential represents a pair of sine and cosine waveforms of identical frequency, Gabor (1946) showed that the shape of the signal for which the product of uncertainties assumes the smallest possible value is the complex exponential (i.e., sine and cosine functions) modulated by a Gaussian envelope or window. These functions will be referred to here as Gabor elementary functions (see Figure 11 for examples of spatial Gabor functions).

There are also other reasons why the Gaussian is particularly attractive and useful as a window function. For instance, it is a smooth function, which is advantageous when derivative operations are required. Also, the family of Gabor elementary functions (GEFs) resembles a useful set of basis functions known as the Hermite polynomials which are generated from a single Gaussian by a sequence of derivative operations, and which are orthogonal with respect to the Gaussian weight function (Kaplan, 1952). Further, the Gaussian of two independent variables is *separable* in both the spatial and spatial frequency domains, which means that it can be expressed as the product of two one-dimensional functions. One-dimensional functions are obviously easier to work with, especially when determining the set of biorthogonal auxiliary functions. The Gaussian is also unique in that it is *selfsimilar* in the spatial and spatial frequency domains, which means that it remains a Gaussian when transformed from one domain to the other. Separability and selfsimilarity are properties of the Gaussian which are shared by no other effectively localized function.

As was noted earlier, it appears to be desirable to combine the spatial and spatial-frequency approaches for analyzing and synthesizing visual images. Thus the question arises as to whether there is an optimal way to represent images in the combined space. This problem has long been appreciated in the area of audition and speech analysis where spectrograms are used to perform spectral analysis within a sliding window of limited duration (i.e., short-term spectral analysis). It is well known that sounds are analyzed by the ear into frequency bands and in fact the sounds that we hear as speech are generated by a relatively small number of such elements, called formants, which are modulated in time (Flanagan, 1965). Recent analysis of the responses of cells in the visual cortex (Daugman, 1985; MacKay, 1981; Marcelja, 1980; Pollen & Ronner, 1983), as well as psychophysical experiments concerned with specifying the luminance distributions which the eye sees best (Watson, Barlow, & Robson, 1983), and the interpretation of such data in the context of image representation in vision (Zeevi & Porat, 1984), suggest that, analogous to the auditory system, the visual system may extract "visual formants" having the form of Gabor functions. Although the total number of such image-forming components per characteristic unit area (which increases in size as a function of eccentricity according to some power law) is much larger than the number of speech-forming components, it may be as small as 4-7 (Watson & Robson, 1981; Wilson & Bergen, 1979) for a given orientation. There are about 15 such characteristic orientations in vision. Therefore, the total number of localized frequency components per unit area is about 100 pairs of cells, with the members of each pair related in quadrature phase.

(a)

(b)

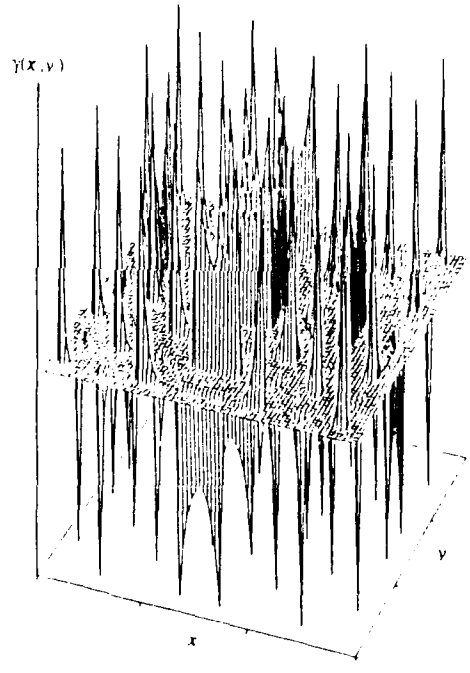


Figure 11. Examples of Gabor Functions and of an Auxiliary Function. (a) Examples of luminance distributions in the form of symmetrical (i.e., cosine component) Gabor functions. Note that the resulting images can vary in position, spatial frequency, orientation, effective width, modulation, and phase. (b) An example of a two-dimensional auxiliary function which is biorthogonal to the Gaussian window of any of the Gabor functions.

Image Representation Using Gabor Functions

Because the spatial and spatial frequency variables described above are complementary quantities, the fundamental principle of uncertainty of signal r· :esentation imposes basic constraints on the structure of elementary functions that can be realized, and hence employed, in any type of image representation. Considering the combined frequency-position space, the most widely used sets of functions are comprised of singular functions (i.e., δ-functions) presented either along the spatial axis or along the spatial frequency axis (where they are also referred to as spectral lines or harmonic functions). A δ-function in either domain implies a function of infinite extent along the complementary axis (Figure 12a)--a condition which is not realizable in practice. As discussed earlier, we seek general functions which are confined in the combined frequency-position space in the sense of being limited in their *effective* (2nd moment) spatial spread and spectral bandwidth. It can be shown that the spatial and spatial-frequency singular functions are the limiting cases of the inherent trade-off that exists between the effective spatial width and the effective spectral width of all possible elementary functions presented in the combined space--in fact, the δ-function is mathematically defined as the limit of a sequence of Gaussians.

To gain some insight with regard to the properties and intrinsic trade-offs characteristic of the Gabor scheme, and for the sake of clarity, we first present the formalism in the context of one-dimensional functions which may be thought of as image crosscuts. Let $g(x)$ be a normalized window function centered at the origin. The localized elementary function of order (m,n) is then defined by:

$$f_{mn}(x) \stackrel{\Delta}{=} g(x - mD) \cdot \exp(jn\omega_0 x) \qquad (1)$$

where m,n are integers, representing the position and frequency numbers, respectively, and $WD \leq 2\pi$. The harmonic function $f_{mn}(x)$ is centered at $(\omega=nW, x=mD)$ in the combined frequency-position space, and the parameters W and D determine how the rectangular Gabor-sampling grid is tessellated (Figure 12b-e). As noted earlier, the choice of a Gaussian for $g(x)$ minimizes the effective area of support (represented by the ellipses in Figure 12a) in the positional-spectral plane compared to the so-called joint entropy achieved by any other window function. This optimal characteristic is, in fact, the main and important advantage of the Gabor elementary functions (GEFs) compared to other localized elementary functions (e.g., those windowed by a squared pulse, one cycle of raised cosine, etc.).

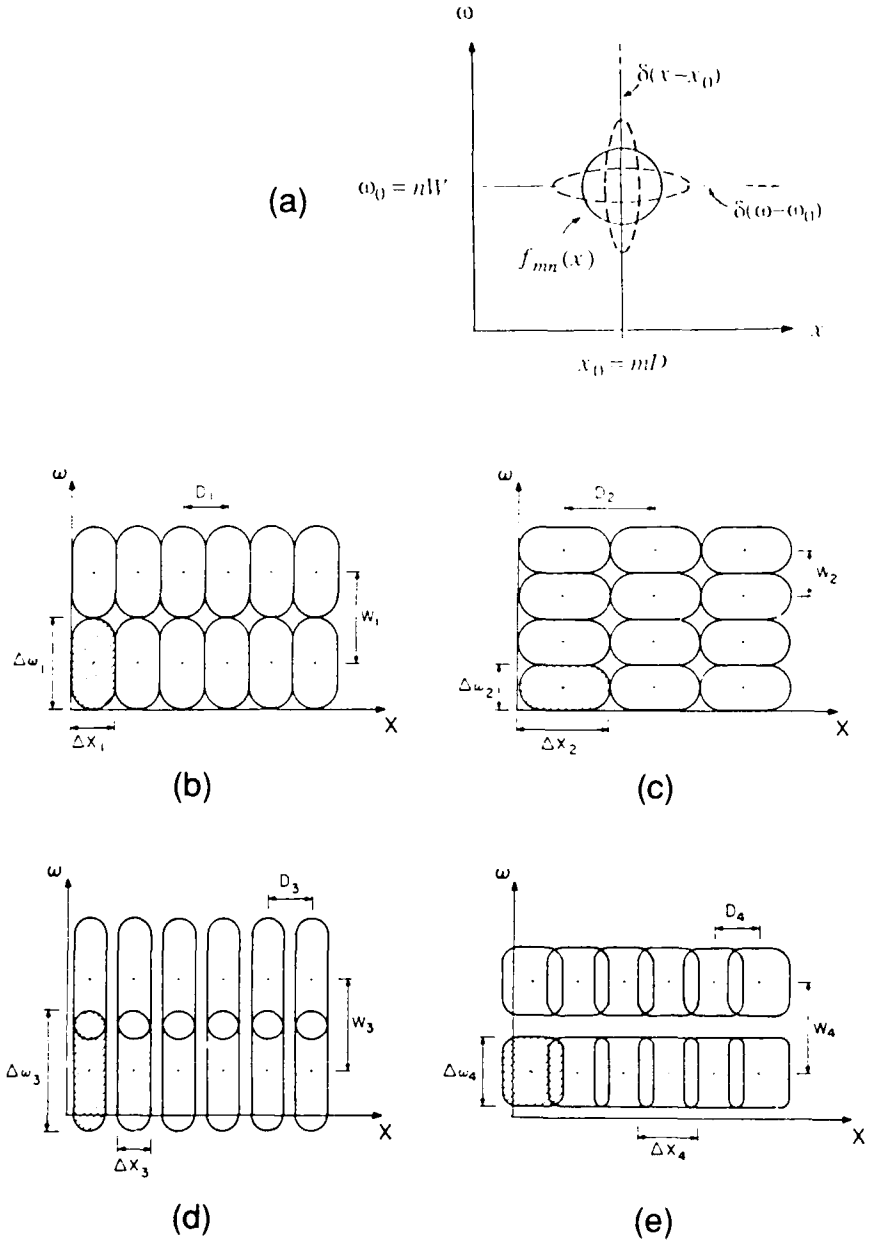


Figure 12. Space/Frequency Trade-Off and Examples of Various Tessellation Schemes. a) Representation of a one-dimensional signal in the combined frequency-position space. The vertical and horizontal lines represent δ-functions along the spatial axis [labelled $\delta(x-xo)$, and representing an infinitely narrow grating], and along the frequency axis [labelled $\delta(\omega-\omega o)$, and representing an infinitely wide grating]. The ellipses, which represent the effective band-area, illustrate that restricting either the spatial or spectral extent of the image results in a concomitant increase in the other dimension. This trade-off is a direct consequence of the basic principle of uncertainty of signal representation. Two of the many possible optimal tessellation schemes satisfying the condition $WD=2\pi$ (i.e., $W_1D_1=W_2D_2$) are shown in (b) and (c). An example of Gabor-space oversampling (i.e., $WD\leq 2\pi$) is shown in (d).

If the condition of optimal information cell size, $WD=2\pi$, is satisfied, the Gabor space is properly sampled, and the set of functions $\{f_{mn}\}$ is complete (Higgins, 1977). Thus, a given one-dimensional crosscut of an image, $\phi(x)$, can be expressed by these elementary functions (Figure 11a), using a set of the corresponding weighting coefficients $\{a_{mn}\}$ describing the relative contribution of each GEF:

$$\phi(x) = \sum_{m=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} a_{mn} \cdot f_{mn}(x)(x) \qquad (2)$$

However, because GEFs are not orthogonal, the analytic formalism for calculating the coefficients employs an auxiliary function $\gamma(x)$ (Bastiaans, 1981; see Figure 11b). This function, which is biorthogonal (Higgins, 1977) in a certain sense to $g(x)$, can be found by solving the kernel of the weighted inner product of the Gaussian and the auxiliary function. In view of the duality between $\gamma(x)$ and $g(x)$, their roles in the forward and inverse transformations can be interchanged. This observation is important for the understanding of the scheme and its implementation in image representation and generation. It implies that either the Gabor elementary functions or the corresponding auxiliary functions can be used in image decomposition for the sake of obtaining the templates of image components (objects). If the auxiliary functions are used in the analysis of images, then the Gabor elementary functions are used in the synthesis (generation) of images, and vice versa (see also previous discussion and Figure 10).

The finite set of expansion coefficients $\{a_{mn}\}$ provides a compact representation of an image crosscut. Graphically two maps of coefficient distributions are needed for a complete definition of an image crosscut--one for the real part, the other for the imaginary part (Figures 13b and 13c). Because the expansion coefficients fully describe an image crosscut (and in the two-dimensional case, which cannot be depicted graphically, they represent an image), they can be considered as the signature of an image in its Gaborian representation.

The basic trade-off between the effective spatial width and the effective spectral width permits the selection of one out of many (theoretically infinite) possible tessellation schemes appropriate for the space confined by the global effective spatial extent and effective frequency band. Thus, the finite scheme requires a fixed number of Gabor components, but permits preselection of any desired number of spectral (Gabor) components for spanning a global (effective) frequency bandwidth. However, according to Equation (2), such a finite scheme affords only an approximate representation or reconstruction of a given signal.

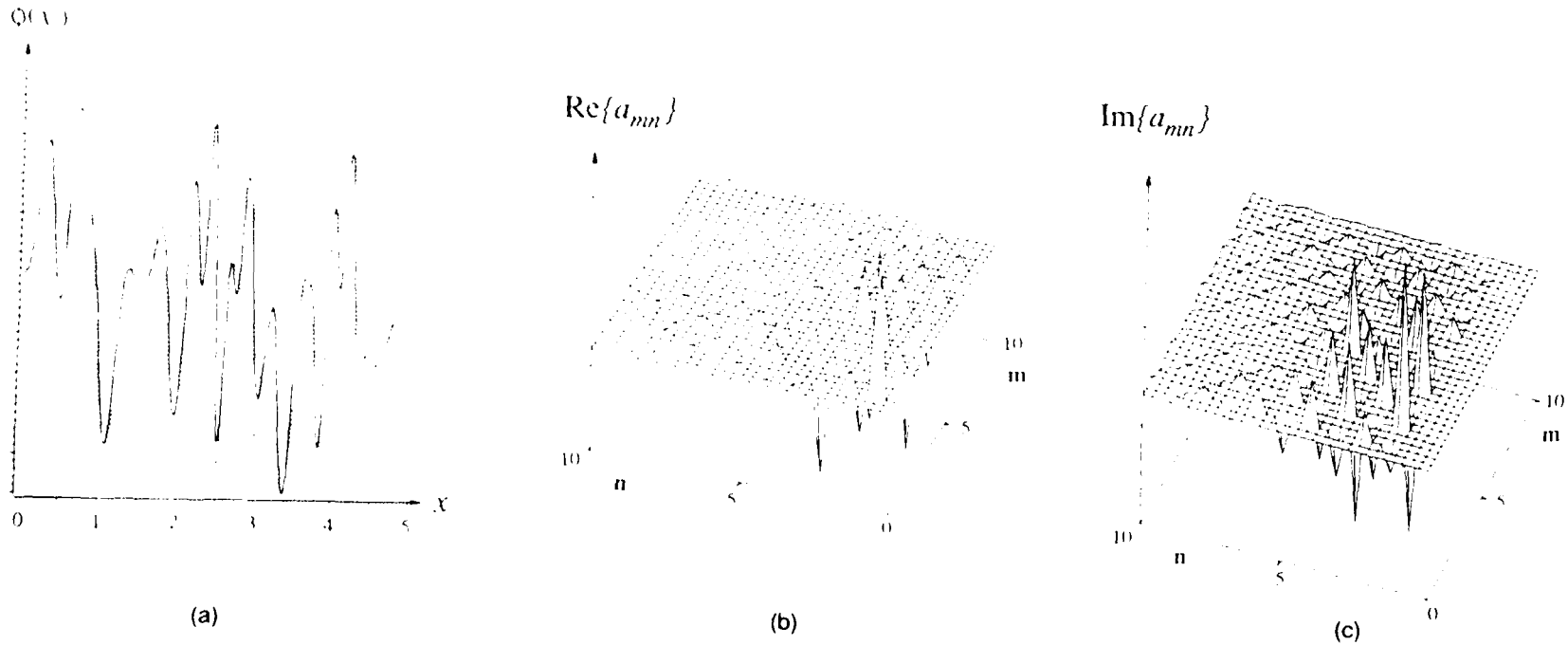


Figure 13. Reconstruction of a Signal From Its Gabor Components. a) An example of decomposition of a one-dimensional signal (representing an image crosscut). The signal is shown by the continuous line, and its reconstructed version, using six GHFs per position, is shown by the dashed line. The spatial coordinate is normalized with respect to D. The distribution of real and imaginary components of the Gabor coefficients are shown in (b) and (c), respectively.

In the case of a two-dimensional scheme, the coefficient space becomes four-dimensional: (x,y) for position and (ω_x,ω_y) for frequency (or, alternatively, $r,\theta,\omega_r,\omega_\theta$, in a polar coordinate system). It can be shown (Porat & Zeevi, 1988) that the two-dimensional representation of a signal $\phi(x,y)$ is given by:

$$\phi(x,y) = \sum_{m_x n_x m_y n_y} a_{m_x n_x m_y n_y} \cdot f_{m_x n_x m_y n_y}(x,y) \tag{3}$$

where a two-dimensional GEF (Figure 11) of order (m_x,n_x,m_y,n_y) is defined by:

$$f_{m_x n_x m_y n_y}(x,y) \stackrel{\Delta}{=} g(x - m_x D_x, y - m_y D_y) \cdot \exp(i n_x W_x x + i n_y W_y y) \tag{4}$$

with the separable Gaussian window function:

$$g(x,y) = g_x(x) \cdot g_y(y). \tag{5}$$

It is required that both one-dimensional window functions $g_x(x)$ and $g_y(x)$ be normalized (of unit energy), and that the conditions of proper information cell size, $W_x D_x \leq 2\pi$ and $W_y D_y \leq 2\pi$, be satisfied.

To calculate the coefficient set $\{a_{m_x n_x m_y n_y}\}$, a two-dimensional auxiliary function (see Figure 11) is employed. Due to the separability of $g(x,y)$, and to the duality of the $g(x)$ and $\gamma(x)$ functions, $\gamma(x,y)$ is also separable. This observation simplifies the extension of the Gabor scheme into two-dimensional (or higher dimensional) systems (Porat & Zeevi, 1988). Using the auxiliary function $\gamma(x,y)$, the coefficients $(a_{m_x n_x m_y n_y})$ are calculated by:

$$a_{m_x n_x m_y n_y} = \int\int \phi(x,y) \cdot \gamma^*(x - m_x D_x, y - m_y D_y) \cdot \exp(-i n_x W_x x - i n_y W_y y)\, dx\, dy \tag{6}$$

For the purpose of image analysis and computer image generation using a system which implements some type of an area of interest (AOI) with eccentricity-dependent sampling and processing, we represent the Gabor scheme in polar coordinates (r,θ). An image $\phi(x,y)$ may be expressed by:

$$\phi(x,y) = \sum_{m_r n_r m_\theta n_\theta} a_{m_r n_r m_\theta n_\theta} \cdot g\left| \sqrt{x^2+y^2} - m_r D_r,\ \tan^{-1}(y/x) - m_\theta D_\theta \right| \cdot \exp\left| i n_r W_r \sqrt{x^2+y^2} + i n_\theta W_\theta \cdot \tan^{-1}(y/x) \right| \tag{7}$$

with the coefficients calculated similarly to those expressed in cartesian coordinates:

$$a_{m_r n_r m_\theta n_\theta} = \int_{\theta=0}^{2\pi} \int_{r=0}^{r_{max}} \phi(r\cos\theta, r\sin\theta) \cdot \gamma^*(r - m_r D_r, \theta - m_\theta D_\theta) \cdot exp(-in_r W_r r - in_\theta W_\theta \theta) dr d\theta \quad (8)$$

Although the image is given in cartesian coordinates, the processing takes place in a polar coordinate system where inhomogeneity is readily incorporated along the r-axis. The image is then encoded by elementary functions representing the parameters of position ($m_r d_r, m_\theta d_\theta$) and spatial frequency ($n_r W_r, n_\theta W_\theta$). This type of cartesian-to-polar coordinate transformation is in accord with the global complex logarithmic mapping (representation) which facilitates certain types of geometric manipulation (Weiman & Chaikin, 1979), and which has been discussed, in the context of human vision, by Schwartz (1980).

Variable Resolution Using Nonuniform Gabor Sampling

The design of the human visual system itself suggests a method for implementing sufficiently high resolution over a wide field of view. The visual system is spatially inhomogeneous in that only a small area near the center of the retina is sensitive to fine spatial detail, and in that the rate of both spatial sampling and processing decreases in all directions toward the visual periphery (Kronauer & Zeevi, 1985; Schwartz, 1980). Recognizing this property, flight simulators are now being designed to provide variable resolution either by partitioning the image into high- and low-resolution subfields such that a small, high-resolution portion of the display is always allocated to that portion of the image being fixated by the operator (Fischetti & Truxal, 1985), or by optically distorting the display such that relatively more raster lines appear in the vicinity of the operator's fixation point (Diehl, 1976). We describe here a further refinement of the variable resolution concept whereby visual images are generated using elementary functions having the form of luminance distributions to which the human visual system is most sensitive (Watson *et al.*, 1983), and in combinations that reflect the most recent data on the changes in visual sensitivity across the retina (JOSA, 1987).

We now proceed to incorporate into the scheme the capability of representing or generating an image by a set of Gabor elementary functions tessellated along a nonuniform Gabor-sampling grid (Figure 14). The basic idea is to implement, in computer-generated imagery (CGI) for flight simulators, a finite Gabor scheme wherein the Gabor sampling rate and the local bandwidth vary as a function of the distance from a focal point to match the characteristics of human vision as a function of eccentricity (Geri, Lyon, & Zeevi, 1989; JOSA, 1987). The result will be an image with high spatial resolution, and also widest spatial-frequency bandwidth, near the center of the visual field, and decreasing resolution (and spatial-frequency bandwidth) as a function of eccentricity (Figure 15). Such a system can, with limited channel capacity and limited computational resources, produce imagery of high perceptual fidelity over a wide field

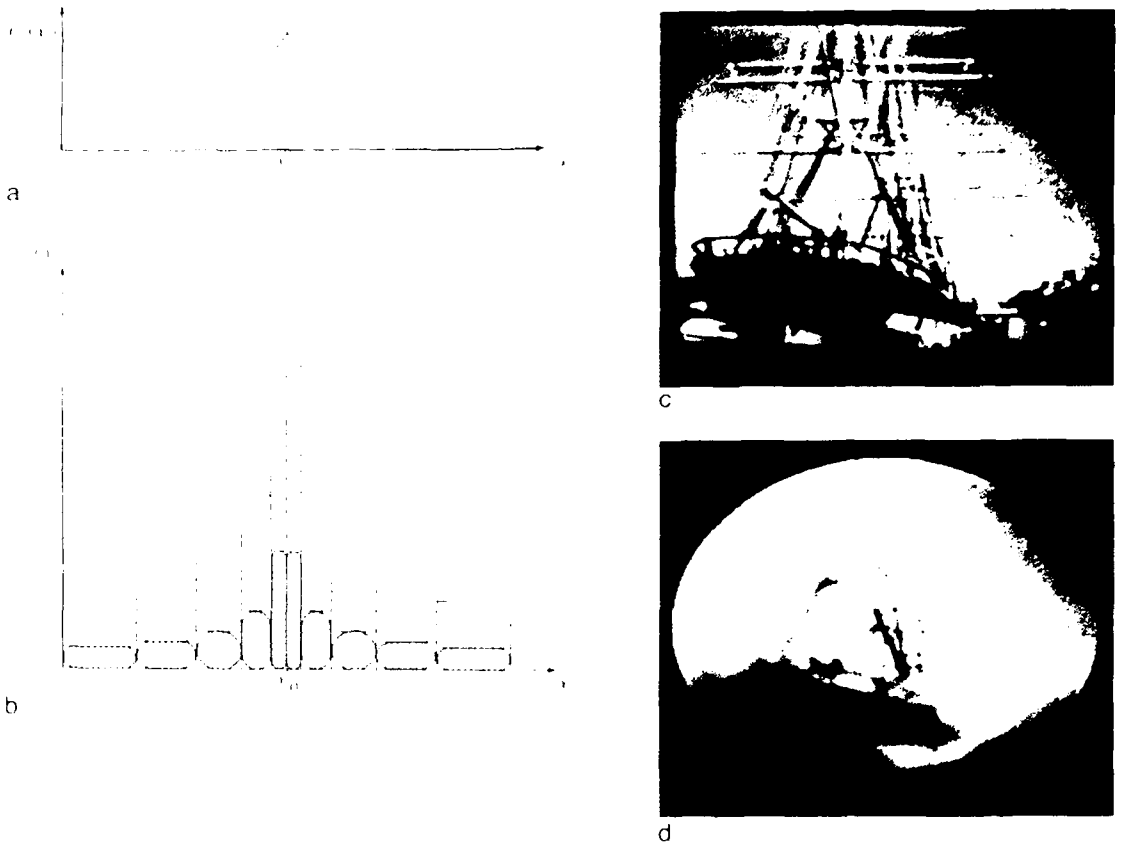


Figure 14. Nonuniform Sampling and Tessellation and a Resulting Variable Resolution Image. (a),(b) the characteristics of one possible nonuniform Gabor-sampling scheme. An example of an image (c) and its reconstruction (d) using nonuniform sampling.

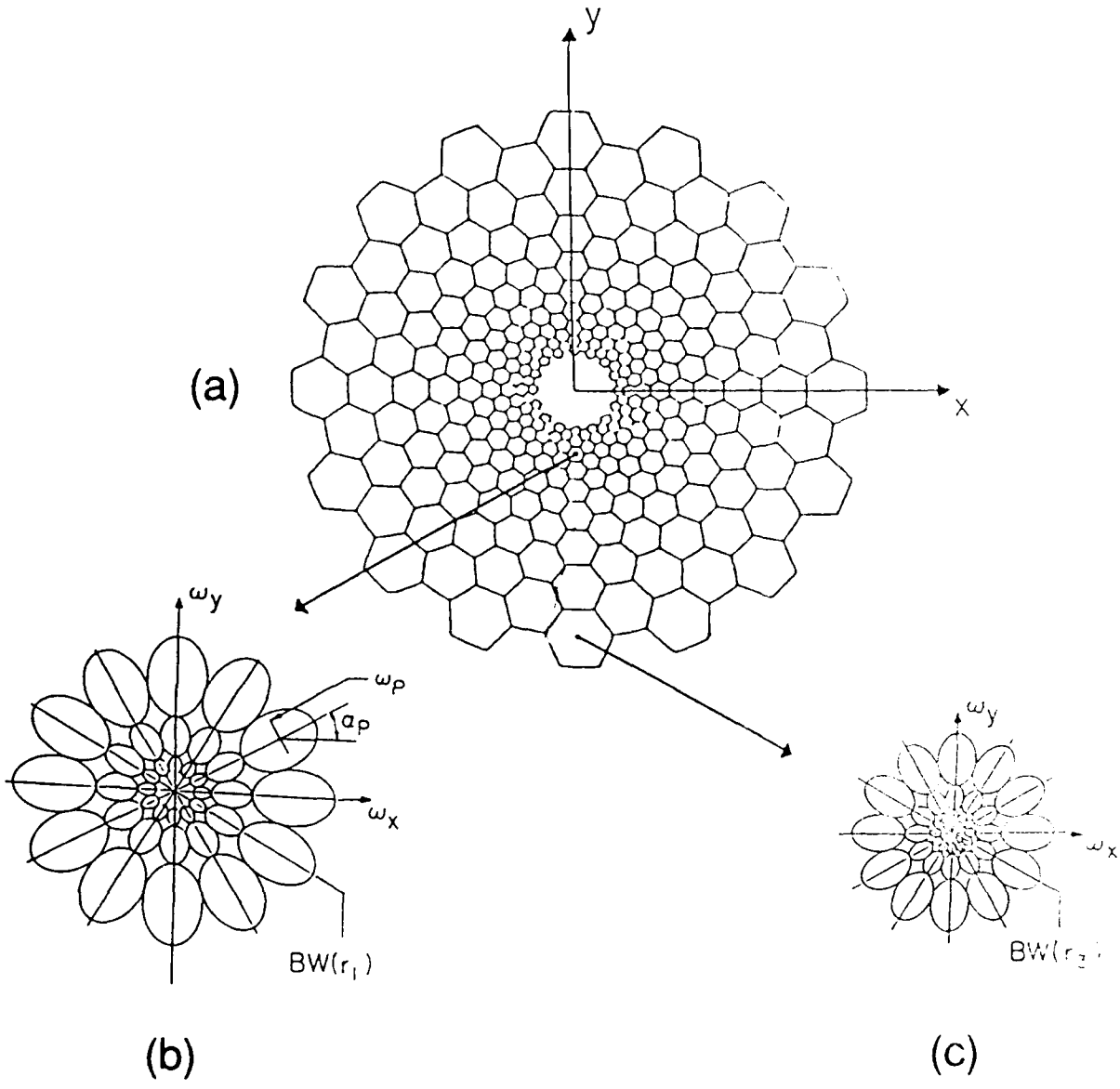


Figure 15. Schematic Representation of Position-Dependent Sampling, Which Shows the Concomitant Space/Frequency Trade-Off. (a) A schematic representation of position-dependent sampling rate most appropriate for gaze-slaved, computer-generated imagery. The area, around the fixation point, which has the highest (and fixed) sampling rate has been left blank for clarity. Each hexagon represents a "Nyquist cell" of information. Shown in (b) and (c) are the spatial frequency representations of the components which are added together at two chosen spatial locations.

of view. It should be noted, however, that the implementation of such a scheme in a flight simulator, or any other system of displayed information, requires continuous measurement of eye position so that the focal point of the displayed information and the observer's point of gaze may be kept coincident. The technology for gaze position measurement is available and has been implemented in the first generation of helmet-mounted-display flight simulators (Fischetti & Truxal, 1985; Robinson, Thomas, & Wetzel, 1989; Williams, Komoda, & Zeevi, 1987).

A system characterized by a position-dependent sampling rate cannot be described by a transfer function (or modulation transfer function) because the impulse response and its transform are strictly applicable only to linear, position-invariant (i.e., spatially uniform) systems. Instead, the concept of local bandwidth (Horiuchi, 1968) must be invoked. In the generalized Gabor scheme, which is characterized by an infinite number of GEFs per Gabor sampling-point, the local bandwidth is theoretically infinite everywhere. In the finite scheme, however, the local bandwidth is inversely proportional to the size of the corresponding information cells, examples of which are depicted by the hexagons in Figure 15.

Given a sampling rate function corresponding to the local density of information in the inhomogeneous system (Figure 12), a distortion function, $S(x)$, can be defined as a functional of the sampling rate. Such a distortion function (see Figure 14a) may, for example, correspond to the cortical magnification factor (cf., Kronauer & Zeevi, 1985; Schwartz, 1980). Having defined the distortion function $S(x)$, the set of expansion coefficients, $\{a_{mn}\}$, corresponding to the inhomogeneous system, are determined by:

$$\tilde{a}_{mn} = \int \phi[S^{-1}(x_d)] \cdot \gamma^*(x_d - mD) \cdot exp(-inh' x_d) dx_d \qquad (9)$$

and the image crosscut is represented accordingly by:

$$\phi(x) = \sum_m \sum_n \tilde{a}_{mn} \cdot g[S(x) - mD] \cdot \exp[inh' \cdot S(x)]. \qquad (10)$$

(For details, see Porat & Zeevi, 1988.) In this scheme the coefficients are calculated using a distorted version of the auxiliary function $\gamma(x)$, and the signal is represented and reconstructed by distorted Gabor functions. This has to be taken into consideration in the design of the special-purpose system (see below) to be used for generating visual imagery using the Gabor approach.

Nonuniform sampling using conventional procedures does not necessarily permit lossless reconstruction because it may not satisfy basic sampling and informational constraints. Further, only for limited cases do there exist procedures for reconstructing an image from nonuniformly spaced samples. In the Gabor case, however, there exist degrees of freedom which permit nonuniform sampling along one coordinate, and which determine in turn the nonuniform sampling along the complementary axis. This is the essence of the [illegible] f between frequency

bandwidth and effective positional spread, which together with the condition of proper sampling (i.e., WD=2π) determine the tessellating scheme. For any such nonuniform Gabor-sampling scheme, the original image can be reconstructed using the entire set (theoretically infinitely large) of elementary functions. The effects of nonuniform sampling become apparent if a finite (relatively small) set of components is used in the reconstruction. For example, if only three frequency components are used per Gabor-sampling position, the "local bandwidth" decreases progressively as a function of the distance from the center (see Figure 14b). This, of course, affects the fidelity of the image as a function of the distance from its center, resulting in the so-called variable resolution image. In the finite scheme of image representation, image quality is related to the effective local bandwidth which can be selected to match visual system characteristics as they vary with eccentricity. In the technique described here, we use two-dimensional GEFs positioned at various locations in the field. Such operators extract (in the case of image analysis) localized 2-D frequency signatures. Because the Gabor operators, like the Fourier, come in pairs with 90-degree phase shift (i.e., sine and cosine functions), the ratio of responses to such a pair extracts the relevant phase information. In the case of image synthesis, such as required in computer image generation for flight simulators, the combination of amplitudes of a sufficient number of pairs of such components can generate any local structure and/or global image. For this reason Gabor operators can be useful in a variety of applications in the field of image science.

In using such a set of GEFs for either image analysis or synthesis, consideration must be given to how many of them are required to generate or represent a given typical image. Here, of course, we are confronted with the problem of determining just what constitutes a perceptually acceptable image. In fact, this point touches upon the definition of image structure, which is one of the most difficult issues in image understanding. In Figure 14b and 14c, we presented an original image and its variable resolution reconstruction. The latter is obviously "lossy" (using the terminology of signal processing) in that some of the information in the original image does not exist in the reconstructed image. However, to an observer who is positioned at the proper distance from the images (obviously, the images would have to be magnified beyond the present page size to match this distance), the images will appear similar, provided that the display system is slaved to eye position.

Synthesis of Fully Textured Images Using Gabor Functions

Several practical problems must be solved before the Gabor approach can be used routinely to produce computer-generated images (CGI). First, the Gabor approach (or, for that matter, any approach) requires the development of a suitable image database. To efficiently exploit the advantages inherent in the Gabor approach requires that complex, fully textured image templates be available for incorporation into the simulated visual scene. Second, provision must be made for manipulating simulated objects via translation, rotation, slanting, and change of size. Third, algorithms and techniques must be developed for gradually changing the nature and amount of texture in the simulated image. And finally, the extensive computation associated with generating Gabor components requires that special-purpose hardware be designed to

provide these components which may then be combined to form the simulated image. Although only partial solutions to these problems exist at present, we are able to offer some general observations and suggestions.

Considering first the design of image databases and the manipulation of simulated objects, we face a problem similar to that encountered in implementing any computer graphics technique which uses a set of image primitives; namely, determining how to combine the primitives to produce the required imagery. If, for example, one defines a set of GEFs which are required to adequately represent an object in the visual scene--a task that can be accomplished using Gabor image analysis--then how can the set of components be manipulated to generate the transformations required due to the movement of the object with 6 degrees of freedom? We reiterate in this context that the Gabor approach combines the advantages offered by a purely spatial approach, in which translation is easily performed, and by a frequency-domain (i.e., Fourier-like) approach whereby changes in size are easily obtained. However, the remainder of the transformations, which are required to simulate change in position and orientation with 6 degrees of freedom, cannot at present be fully implemented by direct manipulation of the sets of coefficient-templates forming the database. Some changes in object and/or terrain orientation (slant) can be incorporated by manipulating the orientation of the 2-D sinusoids and the aspect ratio of the Gaussian windows.

Another practical difficulty in producing conventional CGI involves depicting differences in texture associated with various (usually extensive) objects in the visual field such as terrain, forests, lakes, and the like. Previous work has demonstrated the efficiency of frequency analysis in representing images which appear textured (Kronauer, Zeevi, & Daugman, 1982). More recently, it has been shown that GEFs are more suitable for such an analysis because they are localized and thus able to handle nonuniform textures (Porat & Zeevi, 1989). Similarly, it is possible to synthesize nonuniform, textured images using a relatively small number of Gabor components (Zeevi & Porat, 1988). There is a great deal of redundancy in the structure of images, and those images are processed by a human visual system that is highly nonlinear. Thus, Kronauer *et al.* (1982) were able to show that a small cluster of frequency components properly distributed over the 2-D spatial-frequency space gives rise to a percept very similar to that induced by 2-D bandlimited noise (cf. Mostafavi & Sakrison, 1976). Whereas the dimensionality of the bandlimited noise is extremely high (i.e., many bits of information are required to specify the image), the perceptually equivalent Gabor-textured image is fully specified by only a few numbers. This example illustrates the potential power offered by the GEF approach to the synthesis of textures for CGI.

The final problem to be considered is the extensive computation which would be required to reconstruct an image from its GEF components in real time, as would be necessary, for example, in generating realistic flight simulator imagery. In order to reduce the computational load to manageable levels, special-purpose hardware and new architectures must be developed for generating the GEF components and combining them into the two-dimensional functions which are in turn combined to produce the simulated image. The general layout of a computer image generating system using an adjustable set of hardware-generated GEFs is presented in Figure 16. For the sake of simplicity, we consider in this diagram a system for generating a

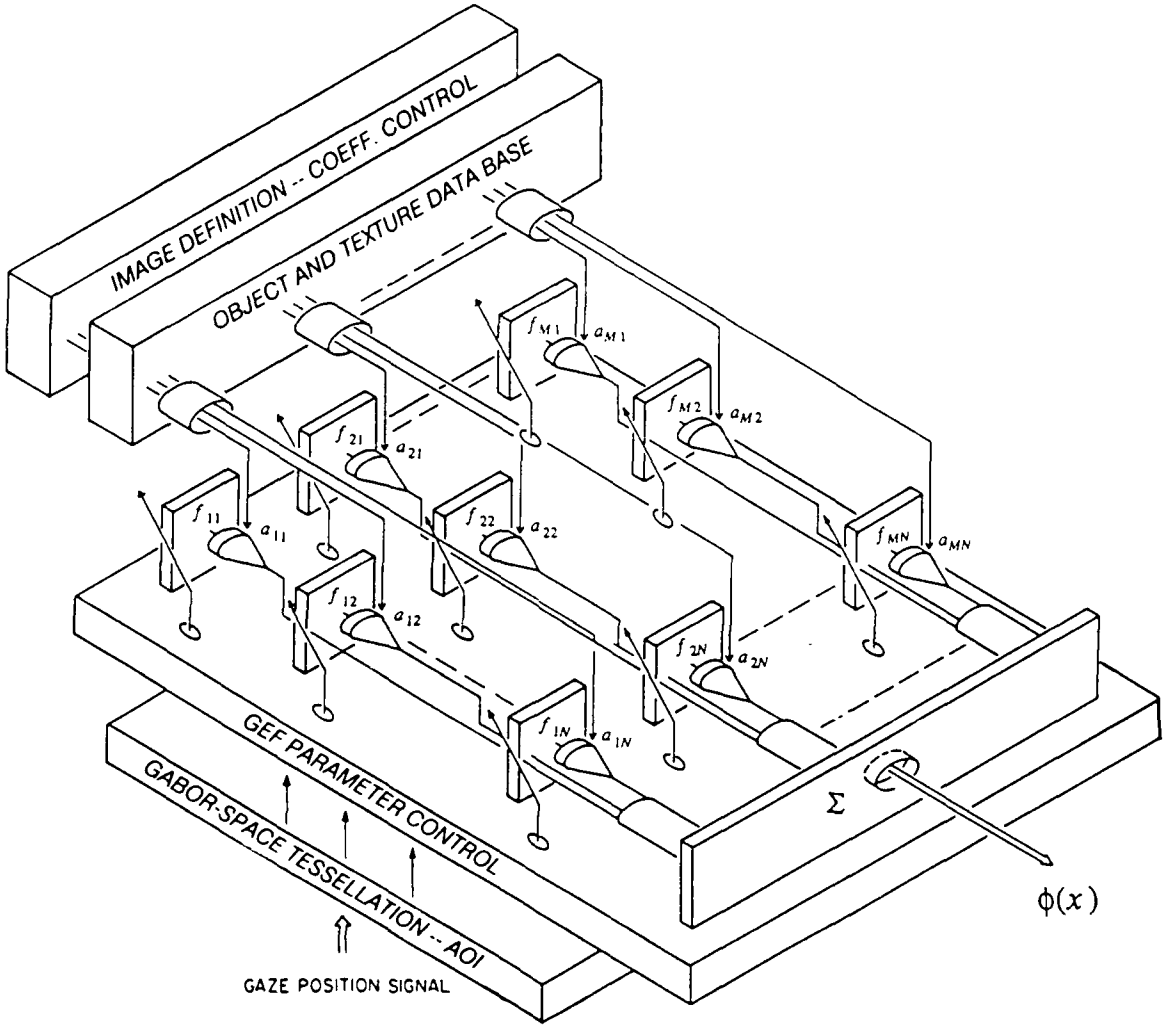


Figure 16. A Possible Hardware Implementation of the Gabor Approach, Which Uses Gaze-Slaved, Computer-Generated Imagery.

one-dimensional signal, $\Phi(x)$, from a set, $\{f_{mn}\}$, of GEFs. In Figure 16, the discrete spatial coordinate (corresponding to $x=mD$) is indicated by the first subscript (from 1 to M) associated with the sets of both the coefficients $\{a_{mn}\}$ and the GEFs $\{f_{mn}\}$. Similarly, the spatial frequency coordinate (corresponding to $\omega=nW$) is indicated by the second subscript (from 1 to N) for both sets. The architecture of the system is highly parallel in that each image is made up of a set of perhaps 10^5 coefficients, each object is made up of a subset of those coefficients, and each coefficient has a direct line to the very large scale integrated (VLSI) circuit module (indicated by the rectangular boxes, labeled f_{mn} in Figure 16) which generates the corresponding GEF (Einziger & Hertzberg, 1986). That is, the set of GEFs is activated and weighted in parallel by a set of lines conveying the coefficients which define the image.

For a given gaze position, which defines the area of interest (Williams *et al.*, 1987), the values f_{mn} are adjusted to give the optimal tessellation for the finite number of GEFs available. Thus, even in the one-dimensional case, four parameters (central spatial position, central frequency, effective spatial spread, and the complementary bandwidth) must be adjusted for each module at each point of gaze. A given image is viewed as being composed of a set of objects and a distribution of texture across the image space. Accordingly, once an image is defined (see first stage at left in Figure 16), an algorithm determines the selection of subsets of coefficients $\{a_{mn}\}$ according to object and texture information stored in the database. As stated earlier, the objects and textures are defined in the database in a generic form only, and so the transformations needed to generate a complete, real-world database must be developed before the proposed CGI system can be realized.

IV. GENERAL DISCUSSION

Over the last several decades, communication engineers and, more recently, computer scientists have been working on interrelated problems, trying to understand the structure of images, and to devise techniques for analyzing, synthesizing, transmitting, and displaying images efficiently. Simultaneously, neuroscientists have made tremendous progress in elucidating the neural mechanisms involved in biological image representation and processing. It is interesting to observe the direct influence of ideas and new findings in one field on new developments in other fields. Although, traditionally, ideas emerging in physics and communication theory influenced new directions of research as well as the models proposed by visual neuroscientists, the trend is reversing and increased mutual interaction is now occurring. This is perhaps not surprising as, in the final analysis, the human observer is often the receiver of displayed information, and thus any communication and display technology will be more effective if it is matched to human capabilities. The interest in biological vision systems--of those involved in the development of advanced visual systems, neurobiological architectures, and machine-vision algorithms--is also due to the recent advances in microelectronics. It is now possible to build miniaturized systems which integrate several hundred thousand, highly interconnected components on a single piece of silicon wafer and to devise algorithms and architectures for parallel processing. All of these capabilities appear to be necessary for building biological-like visual systems.

REFERENCES

Bastiaans, M. J. (1981). A sampling theorem for the complex spectrogram and Gabor expansion of a signal into Gaussian elementary signals. *Optical Engineering*, **20**, 594-598.

Bracewell, R. N. (1986). *The Fourier transform and its applications* (2nd ed.,rev.). New York: McGraw-Hill.

Braddick, O., Campbell, F.W., & Atkinson, J. (1978). Channels in vision: Basic aspects. In *Handbook of sensory physiology*, 8. 3-38. New York: Springer-Verlag.

Campbell, F. W., & Maffei, L. (1974). Contrast and spatial frequency. *Scientific American*, **231**, 106-114.

Campbell, F. W.,& Robson, J. G. (1968). Application of Fourier analysis to the visibility of gratings. *Journal of Physiology*, **197**, 551-566.

Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America*, **2**, 1160-1169.

DeValois, R.L., Albrecht, D.G., & Thorell, L.G. (1982). Spatial frequency selectivity of cells in macaque visual cortex. *Vision Research*, **22**, 545-559.

Diehl, V. E. (1976). A variable acuity remote viewing system. *Proceedings of the IEEE*, National Aerospace and Electronics Conference, p. 663.

Einziger, P. D., & Hertzberg, Y. (1986). On the Gabor representation and its digital implementation. (EE Pub. No. 587), Technion, Haifa, Israel.

Fischetti, M. A., & Truxal, C. (1985). Simulating the right stuff. *IEEE Spectrum*, **22**, 38-47.

Flanagan, J. L. (1965). *Speech analysis synthesis and perception*. New York: Academic Press.

Gabor, D. (1946). Theory of communication. *Journal of the IEE*, **93**, 429-459.

Geri, G. A., Lyon, D. R., & Zeevi, Y. Y. (1989). Discrimination of multicomponent Gabor textures in the central and peripheral visual field. *Investigative Ophthalmology & Visual Science* (Suppl), **30**, 452.

Ginsburg, A. P. (1978). Visual information processing based on spatial filters constrained by biological data. (*AMRL-TR-78-129*), Wright-Patterson AFB, OH: Air Force Aerospace Medical Research Laboratory.

Hartley, R. V. L. (1928). Transmission of information. *Bell System Technical Journal*, 7, 535-563.

Higgins, J. R. (1977). *Completeness and basis properties of sets of special functions*. Cambridge, England: Cambridge University Press

Horiuchi, K. (1968). Sampling principle for continuous signals with time-varying bands. *Information and Control*, **13**, 53-61.

Hubel, D. (1982). Exploration of the primary visual cortex, 1955-78. *Nature*, **299**, 515-524.

JOSA (1987). Variations of visual function across the visual field. Feature section in *Journal of the Optical Society of America A*, **4**, 1478-1703.

Kaplan, W. (1952). *Advanced calculus*. Reading, MA: Addison-Wesley.

Kronauer, R. E., & Zeevi, Y. Y. (1985). Reorganization and diversification of signals in vision. *IEEE Transactions on Systems, Man, and Cybernetics*, **15**, 91-101.

Kronauer, R. E., Zeevi, Y. Y., & Daugman, J. G. (1982). Degree of disorder perceived in images with punctate spectra. *Journal of the Optical Society of America*, **72**, 1798.

MacKay, D. M. (1981). Strife over visual cortical function. *Nature*, **289**, 117-118.

Marcelja, S. (1980). Mathematical description of the responses of simple cortical cells. *Journal of the Optical Society of America*, **70**, 1297-1300.

Mostafavi, H., & Sakrison, D. J. (1976). Structure and properties of a single channel in the human visual system. *Vision Research*, **16**, 957-968.

Nyquist, H. (1924). Certain factors affecting telegraph speed. *Bell System Technical Journal*, **3**, 324.

Oppenheim, A.V., & Lim, J.S. (1981). The importance of phase in signals. *Proceedings of the IEEE*, **69**, 529-541.

Papoulis, A. (1968). *Systems and transforms with applications in optics*. New York: McGraw-Hill.

Pollen, D. A., & Ronner, S. F. (1983). Visual cortical neurons as localized spatial frequency filters. *IEEE Transactions on Systems, Man, and Cybernetics*, **13**, 907-916.

Porat, M., & Zeevi, Y. Y. (1988). The generalized Gabor scheme of image representation in biological and machine vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**, 452-468.

Porat, M., & Zeevi, Y. Y. (1989). Localized texture processing in vision: Analysis and synthesis in the Gaborian space. *IEEE Transactions on Biomedical Engineering*, **36**, 115-129.

Ratliff, F. (1972). Contour and contrast. *Scientific American*, 91-101.

Robinson, R., Thomas, M., & Wetzel, P. (1989). Eye tracker development on the fiber optic helmet mounted display. *Proceedings on Helmet-Mounted Displays*, **SPIE 1116**, 102-108.

Schachter, B. J. (1983). Factors affecting the design of CIG systems. In B. J. Schachter (Ed.), *Computer image generation*.(pp. 27-45). New York: Wiley-Interscience.

Schade, O. H., Sr. (1956). Optical and photoelectric analog of the eye. *Journal of the Optical Society of America*, **46**, 721-739.

Schwartz, E. (1980). Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding. *Vision Research*, **20**, 645-669.

Strang, G. (1986). *Introduction to applied mathematics*. Wellesley, MA: Wellesley-Cambridge Press.

Watson, A. B., Barlow, H. B., & Robson, J. G. (1983). What does the eye see best? *Nature*, **302**, 419-422.

Watson, A. B., & Robson, J. G. (1981). Discrimination at threshold: Labelled detectors in human vision. *Vision Research*, **21**, 1115-1122.

Weiman, C. F. R., & Chaikin, G. (1979). Logarithmic spiral grids for image processing and display. *Computer Graphics and Image Processing*, **11**, 197-226.

Westheimer, G. (1978). Spatial phase sensitivity for sinusoidal grating targets. *Vision Research*, **18**, 1073-1074.

Williams, T., Komoda, M., & Zeevi, Y. Y. (1987). Techniques and methods used in eye-tracking in the fiber-optic helmet-mounted display. *Proceedings of the Image IV Conference*, Phoenix, AZ.

Wilson, H. R., & Bergen, J. R. (1979). A four mechanism model for threshold spatial vision. *Vision Research*, **19**, 19-32.

Zeevi, Y. Y., & Porat, M. (1984). Combined frequency-position scheme of image representation in vision. *Journal of the Optical Society of America A*, **1**, 1284.

Zeevi, Y. Y., & Porat, M. (1988). Computer image generation using elementary functions matched to human vision. In R. A. Earnshaw (Ed.), *Theoretical foundations of computer graphics*. Berlin: Springer.

SOURCE CODE FOR THE PROGRAMS GABORAN.F AND GANAL.F.

The programs presented here implement the Gabor scheme described in Section 3.0. The program GANAL.F was specifically written for use on a laboratory computer with an optimizing compiler (NDP FORTRAN-386).

```
C ===================================================================
C
C Program: GABORAN.F
C Author: M. Porat
C
C This program reads an 256x256 image and calculates its Gabor
C coefficients on a 5x5 Gabor grid (for any other grid size,
C change MXMAX accordingly). The number of spectral components
C is specified by NXMAX (the max is 19). The output
C of the program is the reconstructed image (256 x 256 x 8 bits).
C The coefficients are represented by 8 bits per spectral component.
C-------------------------------------------------------------------
C To use the program without an input file, i.e. to use a synthetic
C image generated by the program, see the instructions under the
C heading ***INPUT IMAGE****.
C-------------------------------------------------------------------
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER ENCODE,DECODE
      REAL*8 GMM(-395:395),GSS(-395:395)
      REAL*8 SNN(-395:395,0:19),CSS(-395:395,0:19)
      INTEGER IN(0:255,0:255),OUT(0:255,0:255),SIZE,RAKP
      INTEGER ARPL(-2:7,0:19,-2:7,0:19),ARMS(-2:7,0:19,-2:7,0:19)
      INTEGER AIPL(-2:7,0:19,-2:7,0:19),AIMS(-2:7,0:19,-2:7,0:19)
      INTEGER ILL(-5:10),IUL(-5:10),MLL(0:5000),MUL(0:5000)
      INTEGER*2 IH(256,256)
      CHARACTER*256 CH(512)
      EQUIVALENCE(CH(1),IH(1,1))
      COMMON PI,DX,DY,STPX,STPY
C***************** D  A  T  A     **********************
      NXMAX=19
      SIZE=255
      COVDEC=4.6D0
      SS=00000.D0
      MXMIN=00
      MXMAX=5
      COVREC=1.3D0
      SAF=0.01D0
      RAKP=0
C*********************************************************
      DX=1.D0
      DY=1.D0
      STPX=1.D0
      STPY=1.D0
      RESOL=SIZE/5.D0
      MYMIN=MXMIN
      NYMAX=NXMAX
      MYMAX=MXMAX
      PI=DATAN(1.D0)*4.D0
      WX=2.D0*PI/DX
      WY=2.D0*PI/DY
      DRX=DX*RESOL
      DRY=DY*RESOL
C***************** INPUT IMAGE **********************
C Instead of the line: IN(I,J)=INT(500.D0*DSIN(2.D0*(J+I)*PI/25.D0)),

FILE=a:gaboran.f     Fri Aug 25 21:03:47 1989     PAGE=1
```

```
C use the lines indicated by CCCC when an input file is to be provided.
C-----------------------------------------------------------------------
CCCC     READ(5,*)((IN(I,J),J=0,255),i=0,255)
      DO 80 I=0,255
      DO 80 J=0,255
      IN(I,J)=INT(500.D0*DSIN(2.D0*(J+I)*PI/25.D0))
CCCC     IN(I,J)=IN(I,J)-128
80    CONTINUE
C************* CALCULATING GMM(I),GSS(I),SNN(I,J),CSS(I,J)*************
998   DO 49 I=-395,395
      GMM(I)=0.D0
      GSS(I)=0.D0
      DO 46 J=0,19
      SNN(I,J)=0.D0
      CSS(I,J)=0.D0
46    CONTINUE
49    CONTINUE
      DO 48 I=-SIZE,SIZE
      DO 41 J=0,19
      SNN(I,J)=DSIN(-1.D0*I/RESOL*WX*J)
      CSS(I,J)=DCOS(-1.D0*I/RESOL*WX*J)
41    CONTINUE
      GMM(I)=GAMX(1.D0*I/RESOL)
      GSS(I)=GX(1.D0*I/RESOL)
48    CONTINUE
      DS=(1.D0/RESOL)**2
C************* LIMITS FOR INTEGRATION AND SUMMATION *******************
      DO 410 I=0,SIZE
      MLL(I)=MXMIN
      MUL(I)=MXMAX
      ML=INT((I/RESOL-COVREC)/DX)
      MU=INT((I/RESOL+COVREC)/DX+0.5D0)
      IF((ML .GT. MXMIN) .AND. (ML .LT. MXMAX)) MLL(I)=ML
      IF(ML .GT. MXMAX) MLL(I)=MXMAX
      IF((MU .LT. MXMAX) .AND. (MU .GT. MXMIN)) MUL(I)=MU
      IF(MU .LT. MXMIN) MUL(I)=MXMIN
410   CONTINUE
      DO 420 I=-5,10
      ILL(I)=0
      IUL(I)=SIZE
      IL=INT((I*1.D0-COVDEC)*DX*RESOL)
      IU=INT((I*1.D0+COVDEC)*DX*RESOL)
      IF((IL .GT. 0) .AND. (IL .LT. SIZE)) ILL(I)=IL
      IF(IL .GT. SIZE) ILL(I)=SIZE
      IF((IU .LT. SIZE) .AND. (IU .GT. 0)) IUL(I)=IU
      IF(IU .LT. 0) IUL(I)=0
420   CONTINUE
C****************** IMAGE DECOMPOSITION *****************
92    DO 10 MX=MXMIN,MXMAX
      INTMX=INT(MX*DRX)
      DO 20 MY=MYMIN,MYMAX
      INTMY=INT(MY*DRX)
      DO 30 NX=0,NXMAX
      DO 40 NY=0,NYMAX


FILE=a:gaboran.f    Fri Aug 25 21:03:47 1989    PAGE=2
```

```
      TRE1=0.D0
      TRE2=0.D0
      TIM1=0.D0
      TIM2=0.D0

      DO 50 INTX=ILL(MX),IUL(MX)
C     DO 50 INTX=0,SIZE
      GMMX=GMM(INTX-INTMX)
      IF(DABS(GMMX).LT.SAF)GOTO 50
      CSSX=CSS(INTX,NX)
      SNNX=SNN(INTX,NX)
      DO 60 INTY=ILL(MY),IUL(MY)
C     DO 60 INTY=0,SIZE
      GG=GMMX*GMM(INTY-INTMY)
      IF(DABS(GG).LT.SAF)GOTO 60
      GIN=GG*IN(INTX,INTY)
      TRE1=TRE1+GIN*CSSX*CSS(INTY,NY)
      TRE2=TRE2-GIN*SNNX*SNN(INTY,NY)
      TIM1=TIM1+GIN*SNNX*CSS(INTY,NY)
      TIM2=TIM2+GIN*CSSX*SNN(INTY,NY)
60    CONTINUE
50    CONTINUE
      ARP=TRE1+TRE2
      ARM=TRE1-TRE2
      AIP=TIM1+TIM2
      AIM=TIM1-TIM2
C=============================================================================
      ARPL(MX,NX,MY,NY)=INT(ARP)
      ARMS(MX,NX,MY,NY)=INT(ARM)
      AIPL(MX,NX,MY,NY)=INT(AIP)
      AIMS(MX,NX,MY,NY)=INT(AIM)
C---ENCODING-----------------------------
      ARPL(MX,NX,MY,NY)=ENCODE(ARP)
      ARMS(MX,NX,MY,NY)=ENCODE(ARM)
      AIPL(MX,NX,MY,NY)=ENCODE(AIP)
      AIMS(MX,NX,MY,NY)=ENCODE(AIM)
40    CONTINUE
30    CONTINUE
20    CONTINUE
10    CONTINUE
      DO 14 NX=0,NXMAX
      DO 14 NY=0,NXMAX
14    CONTINUE
      DO 15 NX=0,NXMAX
      DO 15 NY=0,NXMAX
      DO 15 MX=MXMIN,MXMAX
      DO 15 MY=MXMIN,MXMAX
C---DECODING-----------------------------
      ARPL(MX,NX,MY,NY)=DECODE(ARPL(MX,NX,MY,NY))
      ARMS(MX,NX,MY,NY)=DECODE(ARMS(MX,NX,MY,NY))
      AIPL(MX,NX,MY,NY)=DECODE(AIPL(MX,NX,MY,NY))
      AIMS(MX,NX,MY,NY)=DECODE(AIMS(MX,NX,MY,NY))
15    CONTINUE

FILE=a:gaboran.f     Fri Aug 25 21:03:47 1989     PAGE=3
```

```
12    FORMAT(2413)
222   FORMAT(08109)
C***************** IMAGE RECONSTRUCTION ****************
9993  DO 55 INTX=0,SIZE
      DO 66 INTY=0,SIZE
      TOUT=0.D0
      DO 11 MX=MLL(INTX),MUL(INTX)
      GSSX=GSS(INTX-INT(MX*DRX))
      IF(GSSX.LT.SAF)GOTO 11
      DO 22 MY=MLL(INTY),MUL(INTY)
      GXY2=2.D0*GSSX*GSS(INTY-INT(MY*DRY))
      IF(GXY2.LT.SAF)GOTO 22
      DO 33 NX=0,NXMAX
      CSSX=CSS(INTX,NX)
      SNNX=SNN(INTX,NX)
      DO 44 NY=0,NYMAX
      CXCY=CSSX*CSS(INTY,NY)
      SXSY=SNNX*SNN(INTY,NY)
      CXSY=-CSSX*SNN(INTY,NY)
      SXCY=-SNNX*CSS(INTY,NY)
      ZIHUY=1.D0
      IF( (NX*NY) .EQ. 0)ZIHUY=0.5D0
      IF( (NX+NY) .EQ. 0)ZIHUY=ZIHUY*0.5D0
      TOUT=TOUT+GXY2*ZIHUY*((CXCY-SXSY)*ARPL(MX,NX,MY,NY)
     A                      -(SXCY+CXSY)*AIPL(MX,NX,MY,NY)
     A                      +(CXCY+SXSY)*ARMS(MX,NX,MY,NY)
     A                      -(SXCY-CXSY)*AIMS(MX,NX,MY,NY))
44    CONTINUE
33    CONTINUE
22    CONTINUE
11    CONTINUE
      OUT(INTX,INTY)=INT(TOUT*DS)
66    CONTINUE
55    CONTINUE
C****************** DYNAMIC RANGE SCALING **********************
      IXMAX=-50000
      IXMIN=50000
      DO 27 I=0,255
C     WRITE(6,89)(OUT(I,J),J=0,15)
      DO 27 J=0,255
      IF(OUT(I,J) .GT. IXMAX) IXMAX=OUT(I,J)
      IF(OUT(I,J) .LT. IXMIN) IXMIN=OUT(I,J)
27    CONTINUE
      A=255.D0/(IXMAX-IXMIN)
C****************** OUTPUT PRINTING *********************
      DO 65 I=0,255
      DO 65 J=0,255
      TTOUT=A*(OUT(I,J)-IXMIN)
      IH(I+1,J+1)=ANINT(TTOUT)
65    CONTINUE
      DO 825 I=1,256
      WRITE(6,89)(IH(I,J),J=1,256)
825   CONTINUE
89    FORMAT(1614)
```

```
      END
CC*******************SIMAN(X)*************************
      FUNCTION SIMAN(SX)
      REAL*8 SX,SIMAN
      IF(SX .EQ. 0.D0)SIMAN=0.D0
      IF(SX .GT. 0.D0)SIMAN=1.D0
      IF(SX .LT. 0.D0)SIMAN=-1.D0
      RETURN
      END
CC******************** GX(X)    ************************
      FUNCTION GX(TX)
      REAL*8 GX,TX,PI,DX,DY,STPX,STPY
      COMMON PI,DX,DY,STPX,STPY
      GX=1.189207/DSQRT(DX)*DEXP(-PI*(TX/DX)**2)
      RETURN
        END
CC******************** GAMX(X)  ************************
      FUNCTION GAMX(X)
      REAL*8 GAMX,X,PI,DX,DY,STPX,STPY
      REAL*8 S(0:30),A
      COMMON PI,DX,DY,STPX,STPY
      A=1.D0
      S(0)=.4550866923924480470D+00
      S(1)=-0.8514353735481809360D-03
      S(2)=.2969256977227807090D-03
      S(3)=-0.1933702529933812600D-16
      S(4)=.2351683297719855520D-27
      S(5)=-0.5340910449811621010D-41
      S(6)=.2265160867260152210D-57
      S(7)=-0.1794031632946304100D-76
      S(8)=0.D0
      IF (INT(DABS(X/DX)+0.5D0) .GT. 6) A=0.D0
      GAMX=1.8547122D0*A*DEXP(A*PI*(X/DX)**2)*S(INT(ABS(X/DX)+.5))
     A    /DSQRT(DX)
      RETURN
      END
**********************************************
      FUNCTION ENCODE(X)
      REAL Y,Z,XL4,IX
      REAL*8 X
      INTEGER W,ENCODE
      IX=1.*X
      XL4=11.512925
      XL=5.
C XL IS THE LOWER LIMIT; XL4 MUST BE LOG(500,000/XL)
      ENCODE=255
      IF (ABS(1.*IX).GE.XL) THEN
      Y=ABS(1.*IX)/XL
      Z=LOG(Y)
      W=INT(Z*127./XL4+.5)
      ENCODE=W+INT(127.*(SIMANS(1.*IX)+1)/2.)
      ENDIF
      RETURN
      END


FILE=a:gaboran.f    Fri Aug 25 21:03:47 1989    PAGE=5
```

```
*********************************************
      INTEGER FUNCTION DECODE(IX)
      REAL Y,Z,XL4
      INTEGER W,IX
      XL4=11.512925
      XL=5.
C XL IS THE LOWER LIMIT; XL4 MUST BE LOG(500,000/XL)
      DECODE=0
      IF (IX.LT.255) THEN
      W=IX-INT(127.*(SIMANS(IX-127.)+1.)/2.)
      Z=XL4*W/127.
      Y=EXP(Z)
      DECODE=INT(SIMANS(IX-127.)*Y*XL)
      ENDIF
      RETURN
      END
      FUNCTION SIMANS(SX)
      REAL SX,SIMANS
      IF(SX .EQ. 0.0)SIMANS=0.0
      IF(SX .GT. 0.0)SIMANS=1.0
      IF(SX .LT. 0.0)SIMANS=-1.0
      RETURN
      END
```

```
      PROGRAM Gabor Analysis and Reconstruction

C                     FILENAME:                       ganal.f
C                     AUTHOR:                         M. Porat
C                     PROGRAMMER:                     Christopher Voltz - UDRI
C                     CREATED:                        17Aug88
C                     LAST MODIFIED:          23Aug88
C                     MODIFIED BY:            Christopher Voltz - UDRI
C                     INTERFACE PROTOCOL:     NDP FORTRAN-386
C                     REQUIREMENTS:           None.


C*********************************************************************
C                                                     PROGRAM PARAMETERS
C*********************************************************************

C****           Upper bound on sampling size in X direction
                INTEGER MAX_M_X
                PARAMETER (MAX_M_X = 19)

C****           Upper bound on sampling size in Y direction
                INTEGER MAX_M_Y
                PARAMETER (MAX_M_Y = 19)

C****           Maximum size of image in X direction +1 due to compiler bug
                INTEGER MAX_X_SIZE
                PARAMETER (MAX_X_SIZE = 256)

C****           Maximum size of image in Y direction +1 due to compiler bug
                INTEGER MAX_Y_SIZE
                PARAMETER (MAX_Y_SIZE = 256)

C****           Lower bound on sampling size in X direction
                INTEGER MIN_M_X
                PARAMETER (MIN_M_X = 0)

C****           Lower bound on sampling size in Y direction
                INTEGER MIN_M_Y
                PARAMETER (MIN_M_Y = 0)

C****           Symbolic name for value indicating no error occurred
                INTEGER NO_ERROR
                PARAMETER (NO_ERROR = 0)

C****           Channel number for parameter file
                INTEGER PARAM_FILE
                PARAMETER (PARAM_FILE = 1)

C****           Filename of parameter file
                CHARACTER*64    PARAM_NAME
                PARAMETER (PARAM_NAME = 'ganal.dat')


FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=1
```

```
C****          Channel for screen output
               INTEGER SCREEN
               PARAMETER (SCREEN = 6)


C*********************************************************************
C                                                          DECLARATIONS
C*********************************************************************

C****          Declare external functions
               INTEGER*2       close_bfile, open_bfile
               INTEGER*4       read_bfile, write_bfile
               EXTERNAL                close_bfile, open_bfile, read_bfile, write_bfile

               INTEGER         encode_decode
               EXTERNAL                encode_decode


C****          Declare variables which require a specific size
               CHARACTER*64    filename, current_time
               INTEGER*1       header(0:316), b_image(0:MAX_X_SIZE*MAX_Y_SIZE-1)
               INTEGER*2       error_code, handle
               INTEGER*4       header_size, i, num_bytes


C****          Declare general program variables
               INTEGER double, first_time, i, intergerize_grid, int_m_x, int_m_y
               INTEGER int_x, int_y, j, m_x, m_x_max, m_x_min, m_y, m_y_max, m_y_min
               INTEGER NO, num_freq, variable_grid, x_grid, x_max, x_min, x_size
               INTEGER YES, y_grid, y_size

               REAL            a, covdec, covrec, cxcy, cxsy, drx, dry, ds, dx, dy
               REAL            fres_x, fres_y, gama_x, gama_y, gama_xy, gaus_x, gaus_y
               REAL            gaus_xy, gin, old_n_x, old_n_y, PI, resol_x, resol_y, n_x
               REAL            n_y, saf, sxcy, sxsy, temp_1, temp_2, temp_3, imaginary_1
               REAL            imaginary_2, tout, real_1, real_2, wx, wy, zihuy

C****          Declare and dimension arrays
      INTEGER          imaginary_minus (MIN_M_X:MAX_M_X,MIN_M_Y:MAX_M_Y)
      INTEGER          imaginary_plus  (MIN_M_X:MAX_M_X,MIN_M_Y:MAX_M_Y)
      INTEGER          real_minus              (MIN_M_X:MAX_M_X,MIN_M_Y:MAX_M_Y)
      INTEGER          real_plus               (MIN_M_X:MAX_M_X,MIN_M_Y:MAX_M_Y)
      INTEGER*2        d_low_x                 (MIN_M_X:MAX_M_X)
      INTEGER*2        d_low_y                 (MIN_M_Y:MAX_M_Y)
      INTEGER*2        d_up_x                  (MIN_M_X:MAX_M_X)
      INTEGER*2        d_up_y                  (MIN_M_Y:MAX_M_Y)
      INTEGER*2        in                              (0:MAX_X_SIZE,0:MAX_Y_SIZE)
      INTEGER*2        out                             (0:MAX_X_SIZE,0:MAX_Y_SIZE)
      INTEGER*2        r_low_x                 (0:MAX_X_SIZE)
      INTEGER*2        r_low_y                 (0:MAX_Y_SIZE)
      INTEGER*2        r_up_x                  (0:MAX_X_SIZE)
      INTEGER*2        r_up_y                  (0:MAX_Y_SIZE)
      REAL                     cos_x                   (0:MAX_X_SIZE)

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=2
```

```
      REAL            cos_y                   (0:MAX_Y_SIZE)
              REAL                    gamma_function  (0:8)
              REAL                    gamma_x                 (-MAX_X_SIZE:MAX_X_SIZE)
      REAL                    gamma_y                 (-MAX_Y_SIZE:MAX_Y_SIZE)
      REAL                    gauss_x                 (-MAX_X_SIZE:MAX_X_SIZE)
      REAL                    gauss_y                 (-MAX_Y_SIZE:MAX_Y_SIZE)
      REAL            sin_x                   (0:MAX_X_SIZE)
      REAL                    sin_y                   (0:MAX_Y_SIZE)
      REAL                    sum                             (0:MAX_X_SIZE,0:MAX_Y_SIZE)


C*************************************************************************
C                                                       READ PARAMETERS
C*************************************************************************

              OPEN (UNIT = PARAM_FILE, FILE = PARAM_NAME)
              READ (PARAM_FILE,*) filename, covdec, covrec, dx, dy, x_grid, y_grid
              READ (PARAM_FILE,*) variable_grid, intergerize_grid, YES, NO
              READ (PARAM_FILE,*) saf, double, num_freq

              IF (x_grid .GT. MAX_M_X) THEN
                      WRITE (SCREEN,*) 'Maximum value for M_X exceeded.'
                      WRITE (SCREEN,*) 'Max M_X =', MAX_M_X, ', requested M_X =', x_grid
                      STOP
              ELSEIF (y_grid .GT. MAX_M_Y) THEN
                      WRITE (SCREEN,*) 'Maximum value for M_Y exceeded.'
                      WRITE (SCREEN,*) 'Max M_Y =', MAX_M_Y, ', requested M_Y =', y_grid
                      STOP
              END IF


C*************************************************************************
C                                                               READ IMAGE
C*************************************************************************

C****         Open image file
              filename = filename(1:INDEX(filename, ' ')-1) // '.IMG\0'
              handle = open_bfile(filename, 0, error_code)
              IF (error_code .NE. NO_ERROR) THEN
                      WRITE (SCREEN,*) 'Error opening', filename, '.'
                      WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                      STOP
              END IF


C****         Read image header
              num_bytes = 64
              i = read_bfile(handle, header(0), num_bytes, error_code)
              IF ((i .NE. num_bytes) .OR. (error_code .NE. NO_ERROR)) THEN
                      WRITE (SCREEN,*) 'Error reading header of', filename, '.'
                      WRITE (SCREEN,*) i, 'bytes read.'
                      WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=3
```

```
                STOP
        END IF

C****   Determine the image and comment sizes
        header_size = AND(INT(header(2)),255) + 256*AND(INT(header(3)),255)
        x_size      = AND(INT(header(4)),255) + 256*AND(INT(header(5)),255)
        y_size      = AND(INT(header(6)),255) + 256*AND(INT(header(7)),255)
        IF ((x_size .GT. MAX_X_SIZE) .OR. (y_size .GT. MAX_Y_SIZE)) THEN
                WRITE (SCREEN,*) 'Image is too large.'
                WRITE (SCREEN,*) 'MAX x = ', MAX_X_SIZE, 'MAX y = ', MAX_Y_SIZE
                WRITE (SCREEN,*) 'image x = ', x_size, 'image y = ', y_size
                STOP
        END IF

C****   Read in the comment
        num_bytes = header_size
        i = read_bfile(handle, header(64), num_bytes, error_code)
        IF ((i .NE. num_bytes) .OR. (error_code .NE. NO_ERROR)) THEN
                WRITE (SCREEN,*) 'Error reading comment of', filename, '.'
                WRITE (SCREEN,*) i, 'bytes read.'
                WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                STOP
        END IF
        header_size = header_size + 64

C****   Read in the image
        num_bytes = INT(x_size)*y_size
        i = read_bfile(handle, b_image, num_bytes, error_code)
        IF ((i .NE. num_bytes) .OR. (error_code .NE. NO_ERROR)) THEN
                WRITE (SCREEN,*) 'Error reading', filename, '.'
                WRITE (SCREEN,*) i, 'bytes read.'
                WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                STOP
        END IF

C****   Close the file
        error_code = close_bfile(handle)
        IF (error_code .NE. NO_ERROR) THEN
                WRITE (SCREEN,*) 'Error closing', filename, '.'
                WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                STOP
        END IF

C****   Write image parameters
        WRITE (SCREEN,*) 'Image file:  ', filename(1:INDEX(filename, ' ')-1)
        WRITE (SCREEN,*) 'X_grid_size =', x_grid
        WRITE (SCREEN,*) 'Y_grid_size =', y_grid
        WRITE (SCREEN,*) 'X_size      =', x_size
        WRITE (SCREEN,*) 'Y_size      =', y_size

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=4
```

```
C****           Convert the image from binary to a normal INTEGER format
                WRITE (SCREEN,*) '\n===> Converting image'
                DO 5 i = 0, y_size-1
                        DO 5 j = 0, x_size-1
                                in(i,j) = AND(INT(b_image(j+y_size*i)),255)
5               CONTINUE
                x_size = x_size - 1
                y_size = y_size - 1


C*************************************************************************
C                                                       PERFORM CALCULATIONS
C*************************************************************************

C****           debugging statement to be removed at a later time
                call time(current_time)


C****           Initialize global constants
                PI = 4.0*ATAN(1.0)

                wx = 2.0*PI/dx
                wy = 2.0*PI/dy

                gamma_function(0) =  0.4550866923924480 4700e0
                gamma_function(1) = -0.8514353735481809 3600e-3
                gamma_function(2) =  0.2969256977227807 0900e-8
                gamma_function(3) = -0.1933702529933381 2600e-16
                gamma_function(4) =  0.2351683297719855 5200e-27
                gamma_function(5) = -0.5340910449811621 0100e-41
                gamma_function(6) =  0.2265160867260152 2100e-57
                gamma_function(7) = -0.1794031632946304 1000e-76
                gamma_function(8) =  0.0000000000000000 0000e0


C****           Initialize arrays
                WRITE (SCREEN,*) '===> Initializing'
      DO 1200 i = 0, x_size
                DO 1200 j = 0, y_size
                                out(i,j) = 0
                                sum(i,j) = 0.0
1200  CONTINUE

      m_x_min = 0
      m_y_min = 0
      IF (variable_grid .EQ. NO) THEN
         m_x_max = x_grid
         m_y_max = y_grid
      ENDIF

      first_time = YES


FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=5
```

```
2100  READ (PARAM_FILE,*, END = 2300) n_x, n_y
              WRITE (SCREEN,*) '===> Number of frequencies left to calculate = ',
&                                                  num_freq
              num_freq = num_freq - 1

      IF (intergerize_grid .EQ. YES) THEN
         n_x = ANINT(n_x)
         n_y = ANINT(n_y)
      ENDIF

      IF (variable_grid .EQ. YES) THEN
         m_x_max = MAX(x_grid, NINT(x_grid*ABS(n_x)))
         m_y_max = MAX(y_grid, NINT(y_grid*ABS(n_y)))
      ENDIF

      IF ((double .EQ. YES)  .OR. (first_time .EQ. YES) .OR.
&                (n_x .NE. old_n_x) .OR. (n_y .NE. old_n_y)) THEN

              resol_x = REAL(x_size)/REAL(m_x_max)
              resol_y = REAL(y_size)/REAL(m_y_max)
              drx = resol_x*dx
              dry = resol_y*dy
              ds  = 1.0/(resol_x*resol_y)
              zihuy = 1.0
              IF ((n_x*n_y) .EQ. 0) zihuy = 0.5
              IF ((n_x+n_y) .EQ. 0) zihuy = zihuy*0.5
              fres_x = REAL(x_size)/REAL(x_grid)
              fres_y = REAL(y_size)/REAL(y_grid)


C****         Initialize the arrays sin_x, sin_y, cos_x, cos_y
                      temp_1 = wx*n_x/fres_x
              DO 4100 i = 0, x_size
                      sin_x(i) = SIN(-REAL(i)*temp_1)
                      cos_x(i) = COS(-REAL(i)*temp_1)
4100          CONTINUE

                      temp_1 = wy*n_y/fres_y
              DO 4200 i = 0, y_size
                      sin_y(i) = SIN(-REAL(i)*temp_1)
                      cos_y(i) = COS(-REAL(i)*temp_1)
4200          CONTINUE


C****         Initialize the arrays gamma_x, gamma_y
                      temp_1 = SQRT(dx)
              DO 5100 i = -x_size, x_size
                      temp_2 = i/drx
                      IF (NINT(ABS(temp_2)) .GT. 6) THEN
                      gamma_x(i) = 0.0
                      gauss_x(i) = 0.0
                      ELSE
                                      temp_3 = temp_2**2

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=6
```

```
                gamma_x(i) = 1.8547122/temp_1*EXP(PI*temp_3)*
&                            gamma_function(NINT(ABS(temp_2)))
                gauss_x(i) = 1.189207/temp_1*EXP(-PI*temp_3)
                ENDIF
5100        CONTINUE

                temp_1 = SQRT(dy)
            DO 5200 i = y_size, y_size
                temp_2 = i/dry
                IF (NINT(ABS(temp_2)) .GT. 6) THEN
                gamma_y(i) = 0.0
                gauss_y(i) = 0.0
                ELSE
                                temp_3 = temp_2**2
                gamma_y(i) = 1.8547122/temp_1*EXP(PI*temp_3)*
&                            gamma_function(NINT(ABS(temp_2)))
                gauss_y(i) = 1.189207/temp_1*EXP(-PI*temp_3)
                ENDIF
5200        CONTINUE

C****       Compute the limits for summation for decomposition
            DO 5300 i = m_x_min, m_x_max
                        d_low_x(i) = MAX(0, MIN(INT((i-covdec)*drx), x_size))
                        d_up_x(i)  = MAX(0, MIN(INT((i+covdec)*drx), x_size))
5300        CONTINUE

            DO 5400 i = m_y_min, m_y_max
                        d_low_y(i) = MAX(0, MIN(INT((i-covdec)*dry), y_size))
                        d_up_y(i)  = MAX(0, MIN(INT((i+covdec)*dry), y_size))
5400        CONTINUE

C****       Compute the limits for summation for reconstruction
            DO 6300 i = 0, x_size
                        r_low_x(i) = MAX(m_x_min, MIN(INT((i/resol_x-covrec)/dx), m_x_max))
                        r_up_x(i)  = MAX(m_x_min, MIN(NINT((i/resol_x+covrec)/dx), m_x_max))
6300        CONTINUE

            DO 6400 i = 0, y_size
                        r_low_y(i) = MAX(m_y_min, MIN(INT((i/resol_y-covrec)/dy), m_y_max))
                        r_up_y(i)  = MAX(m_y_min, MIN(NINT((i/resol_y+covrec)/dy), m_y_max))
6400        CONTINUE

C****       Perform image decompositon
                WRITE (SCREEN,*) '     Decomposing image'
            DO 5800 m_x = m_x_min, m_x_max
C                       WRITE (SCREEN,*) m_x, 'out of', m_x_max
                int_m_x = INT(m_x*drx+0.01)
                DO 5700 m_y = m_y_min, m_y_max
                int_m_y = INT(m_y*dry+0.01)
                real_1 = 0.0
                real_2 = 0.0

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=7
```

```
                        imaginary_1 = 0.0
                        imaginary_2 = 0.0
                        DO 5600 int_x = d_low_x(m_x), d_up_x(m_x)
C To use the gaussian rather than the gama function, uncomment next line
C                               gama_x = gauss_x(int_x-int_m_x)/1000
C To use the gamma function rather than the gaussian, uncomment next line
                                        gama_x = gamma_x(int_x-int_m_x)
                                IF (ABS(gama_x) .LT. saf) GOTO 5600
                                DO 5500 int_y = d_low_y(m_y), d_up_y(m_y)
C To use the gaussian rather than the gama function, uncomment next line
C                                       gama_y = gauss_y(int_y-int_m_y)/1000
C To use the gamma function rather than the gaussian, uncomment next line
                                                gama_y = gamma_y(int_y-int_m_y)
                                        gama_xy = gama_x*gama_y
                                        IF (ABS(gama_xy) .LT. saf) GOTO 5500
                                        gin = gama_xy*in(int_x,int_y)
                                        real_1 = real_1+gin*cos_x(int_x)*cos_y(int_y)
                                        real_2 = real_2-gin*sin_x(int_x)*sin_y(int_y)
                                        imaginary_1 = imaginary_1+gin*sin_x(int_x)*cos_y(int_y)
                                        imaginary_2 = imaginary_2+gin*cos_x(int_x)*sin_y(int_y)
5500                            CONTINUE
5600                    CONTINUE
                        real_plus(m_x,m_y)              = encode_decode(real_1+real_2)
                        real_minus(m_x,m_y)             = encode_decode(real_1-real_2)
                        imaginary_plus(m_x,m_y) = encode_decode(imaginary_1+imaginary_2)
                        imaginary_minus(m_x,m_y)= encode_decode(imaginary_1-imaginary_2)
5700                    CONTINUE
5800            CONTINUE


C****           Perform image reconstruction
                        WRITE (SCREEN,*) '      Reconstructing image'
                DO 6800 int_x = 0, x_size
C                               WRITE (SCREEN,*) int_x, 'out of', x_size
                        DO 6700 int_y = 0, y_size
                        tout = 0.0
                        DO 6600 m_x = r_low_x(int_x), r_up_x(int_x)
                                        int_m_x = INT(m_x*drx+0.01)
                                        gaus_x = gauss_x(int_x-int_m_x)
                                                gaus_x = gauss_x(int_x-INT(m_x*drx+0.01))
                                        IF (gaus_x .LT. saf) GOTO 6600
                                        DO 6500 m_y = r_low_y(int_y), r_up_y(int_y)
                                                int_m_y = INT(m_y*dry+0.01)
                                                gaus_y = gauss_y(int_y-int_m_y)
                                                gaus_xy = 2.0*gaus_x*gaus_y
                                                IF (gaus_xy .LT. saf) GOTO 6500
                                                cxcy = cos_x(int_x)*cos_y(int_y)
                                                sxsy = sin_x(int_x)*sin_y(int_y)
                                                cxsy = -cos_x(int_x)*sin_y(int_y)
                                                sxcy = -sin_x(int_x)*cos_y(int_y)
                                                tout = tout+gaus_xy*zihuy
&                                                       *((cxcy-sxsy)*real_plus(m_x,m_y)
&                                                       -(sxcy+cxsy)*imaginary_plus(m_x,m_y)
&                                                       +(cxcy+sxsy)*real_minus(m_x,m_y)

FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=8
```

```
&                                                    -(sxcy-cxsy)*imaginary_minus(m_x,m_y))
6500                                  CONTINUE
6600                  CONTINUE
                      sum(int_x,int_y) = sum(int_x,int_y)+(tout*ds)
6700                  CONTINUE
6800          CONTINUE

         first_time = NO
         old_n_x = n_x
         old_n_y = n_y
      ENDIF
      GOTO 2100

2300  CLOSE (PARAM_FILE)


C**** Dynamic range scaling
              WRITE (SCREEN,*) '===> Scaling'
      x_max = -50000
      x_min = 50000
      DO 4105 i = 0, x_size
              DO 4105 j = 0, y_size
                      out(i,j) = INT(sum(i,j))
                              x_max = MAX(x_max, out(i,j))
                              x_min = MIN(x_min, out(i,j))
4105  CONTINUE
      a = 255.0/(x_max-x_min)


C****         debugging statements to be removed at a later time
              write (*,*) 'start= ', current_time
              call time(current_time)
              write (*,*) 'stop= ', current_time
              write (*,*) 'min=', x_min, 'max=', x_max



C*************************************************************************
C                                                              WRITE OUT IMAGE
C*************************************************************************

C****         Convert image from integer to binary format
              x_size = x_size + 1
              y_size = y_size + 1
              WRITE (SCREEN,*) '===> Converting image to binary format.'
      DO 5105 i = 0, y_size-1
                  DO 5105 j = 0, x_size-1
                 b_image(j+i*y_size) = NINT(a*(out(i,j)-x_min))
5105  CONTINUE

C****         Open file
              WRITE (SCREEN,*) '===> Writing image.'
              filename = filename(1:index(filename, '.')-1) // 'r.img\0'
              handle = open_bfile(filename, 256, error_code)


FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=9
```

```
                IF (error_code .NE. NO_ERROR) THEN
                        WRITE (SCREEN,*) 'Error opening', filename, '.'
                        WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                        STOP
                END IF

C****           Write header
                num_bytes = header_size
                i = write_bfile(handle, header, num_bytes, error_code)
                IF ((i .NE. num_bytes) .OR. (error_code .NE. NO_ERROR)) THEN
                        WRITE (SCREEN,*) 'Error writing header of', filename, '.'
                        WRITE (SCREEN,*) i, 'bytes written.'
                        WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                        STOP
                END IF

C****           Write image
                num_bytes = INT(x_size)*y_size
                i = write_bfile(handle, b_image, num_bytes, error_code)
                IF ((i .NE. num_bytes) .OR. (error_code .NE. NO_ERROR)) THEN
                        WRITE (SCREEN,*) 'Error writing', filename, '.'
                        WRITE (SCREEN,*) i, 'bytes written.'
                        WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                        STOP
                END IF

C****           Close file
                error_code = close_bfile(handle)
                IF (error_code .NE. NO_ERROR) THEN
                        WRITE (SCREEN,*) 'Error closing', filename, '.'
                        WRITE (SCREEN,*) 'DOS error code = ', error_code, '.'
                        STOP
                END IF

        END



C*************************************************************************
C                                                                FUNCTIONS
C*************************************************************************

        INTEGER FUNCTION encode_decode(x)
        REAL x

C       This function is used in computing the arrays imaginary_plus,
C imaginary_minus, real_plus and real_minus.


C****           Define local parameters


FILE=a:ganal.f    Sat Aug 26 03:29:09 1989    PAGE=10
```

```
              REAL                    xl
      PARAMETER       (xl   = 5.0)

              REAL                    xl4
      PARAMETER       (xl4 = 11.512925)

C****         Define local variables
      INTEGER                 encode


      IF (ABS(x) .GE. xl) THEN
              IF (x .EQ. 0.0) THEN
                      encode = NINT(127*LOG(ABS(x)/xl)/xl4)+INT(127*1/2)
                      ELSEIF (x .GT. 0.0) THEN
                              encode = NINT(127*LOG(ABS(x)/xl)/xl4)+127
              ELSE
                      encode = NINT(127*LOG(ABS(x)/xl)/xl4)
              ENDIF
      ELSE
                      encode = 255
              ENDIF

      IF (encode .LT. 255) THEN
                      IF ((encode-127) .EQ. 0) THEN
                              encode_decode = 0
                      ELSEIF ((encode-127) .GT. 0) THEN
                              encode_decode = INT(EXP(xl4*(encode-127)/127)*xl)
              ELSE
                      encode_decode = INT(-EXP(xl4*encode/127)*xl)
              ENDIF
              ELSE
                      encode_decode = 0
      ENDIF

      RETURN
      END
```