November 1989 UILU-ENG-89-2236

COORDINATED SCIENCE LABORATORY
College of Engineering

TOMOGRAPHIC PROCESSING OF SYNTHETIC APERTURE RADAR SIGNALS FOR ENHANCED RESOLUTION

Jerald Lee Bauck

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-89-2236	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION U.S. Army Research Office
6c. ADDRESS (City, State, and ZIP Code) 1101 W. Springfield Ave. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) P.O. Box 12211 Research Triangle Park, NC 27709-2211
8a. NAME OF FUNDING / SPONSORING ORGANIZATION U.S. Army Research Office 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DAAL 03-86-K-0111
8c. ADDRESS (City, State, and ZIP Code) P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
TOMOGRAPHIC PROCESSING OF SYNTHETIC APERTURE RADAR SIGNALS FOR ENHANCED RESOLUTION

12. PERSONAL AUTHOR(S)
Bauck, Jerald Lee

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM _____ TO _____	1989 November	217

16. SUPPLEMENTARY NOTATION The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
tomographic signal processing, imaging, synthetic aperture radar, reflection tomography

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Spotlight-mode synthetic aperture radar imaging is studied from the viewpoint of tomographic signal processing which allows the relaxation of the nearly-universal assumption that plane waves pass over the ground patch. This allows high-quality image reconstruction in the face of arbitrary amounts of wavefront curvature such as would be present when the angle subtended by the ground patch, as seen by the radar, is not small. One such application is wide-area surveillance. A methodology is used which has the benefits of a wideband transmitted signal (impulse) and a sensible simulation. Image reconstruction algorithms are developed for monostatic and bistatic systems. Simulation results using these algorithms compare favorably with baseline simulations which use a more conventional algorithm operating on data which do not embody the effects of wavefront curvature. Comments on system design and computational implementation are made as necessary. A new set of problems which appear to benefit from the tomographic viewpoint is posed. This work may also find applications in some forms of reflection tomography.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

DD Form 1473, JUN 86 *Previous editions are obsolete.*

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

A note from the author

This manuscript was produced as a Postscript file then converted to Portable Document Format in March 2019 from the original FullWrite Professional file which was produced in November 1989. Some minor formatting and typesetting differences are present compared to the original printed version and some cover page annotations are missing but otherwise it is identical to the original which is available as a scanned document from Defense Technical Information Center, `https://apps.dtic.mil/dtic/tr/fulltext/u2/a217178.pdf`.

I can be reached by e-mail at this address after left-shifting each alpha character by one position, e.g., f becomes e: kfssz@cbvdl.ofu.

Jerry Bauck
Tempe, Arizona, USA
March 2019

TOMOGRAPHIC PROCESSING OF SYNTHETIC APERTURE RADAR SIGNALS FOR ENHANCED RESOLUTION

BY

JERALD LEE BAUCK

B.S., Kansas State University, 1977
M.S., University of Illinois, 1979

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1990

Urbana, Illinois

ABSTRACT

Spotlight-mode synthetic aperture radar imaging is studied from the viewpoint of tomographic signal processing which allows the relaxation of the nearly-universal assumption that plane waves pass over the ground patch. This allows high-quality image reconstruction in the face of arbitrary amounts of wavefront curvature such as would be present when the angle subtended by the ground patch, as seen by the radar, is not small. One such application is wide-area surveillance. A methodology is used which has the benefits of a wideband transmitted signal (impulse) and a sensible simulation. Image reconstruction algorithms are developed for monostatic and bistatic systems. Simulation results using these algorithms compare favorably with baseline simulations which use a more conventional algorithm operating on data which do not embody the effects of wavefront curvature. Comments on system design and computational implementation are made as necessary. A new set of problems which appear to benefit from the tomographic viewpoint is posed. This work may also find applications in some forms of reflection tomography.

ACKNOWLEDGEMENTS

I thank Professor Kenneth Jenkins for his guidance and understanding as my academic advisor, and the members of my committee. Thanks also go to Andy Hull, Tom Blazek, Mariesa Crow, Jim Drewniak, and especially Maggie Desenberg.

This work is dedicated to Mom and Dad.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION AND SUMMARY

SINCE the invention of synthetic aperture radar (SAR) in the 1950s, much of the analysis and virtually all of the signal processing for SAR have been based on the assumption that plane waves are emitted by the radar antenna. Even though this is clearly at odds with the nature of waves, the assumption, manifested as an asymptotically true approximation, has sufficed for many system designs. There are some applications, however, in which it is advantageous to relax this assumption. The result of doing so can be higher-quality imagery. It is the main purpose of this dissertation to introduce a set of new imaging problems which arise under the more general conditions and to demonstrate corresponding new signal processing schemes for image reconstruction from SAR data. This chapter will proceed with a brief survey of SAR applications and types and conclude with a more refined statement of the problem that was studied and a summary of results.

1.1 Applications of Synthetic Aperture Radar

Synthetic aperture radar is a microwave imaging technique that can provide high-resolution photograph-like images of the Earth and other objects. It is able to do so at night and through clouds, and thus provides a valuable supplement or replacement to other imaging modes. Additionally, SAR extends the observable region of the electromagnetic spectrum in these applications, providing information that was not available before.

Since the conception of SAR by Carl Wiley at the Goodyear Aircraft Corporation in the early 1950s and simultaneous experiments by researchers at the University of Illinois, followed by data processing realizations in hardware by groups at the University of Illinois (electronic) in early 1953 and the University of Michigan (optical) in the summer of 1953, there have been numerous such radars built to service a variety of applications [1], [2], [3], [4]. These radars have been flown both on aircraft and spacecraft, with the applications differing somewhat in each case. Reference [5] includes a historical perspective.

Airborne SARs are useful in such diverse applications as mapping areas that are covered by rain forests, tracking the formation and movement of ice in the north seas, performing reconnaissance from both piloted aircraft and remotely-piloted vehicles [6], battlefield surveillance (a proposed system would have 1 ft range by 1 ft cross-range resolution [7]), and monitoring oil slicks [8]. More applications can be found in the references cited.

Applications for spaceborne SARs are somewhat different. Results have been obtained in the fields of geology, oceanography, glaciology, and agriculture. Specific applications and scientific subjects studied include geologic mapping, subsurface penetration studies, soil moisture and vegetation mapping including measurement of the clear cutting of tropical forests, land utilization, man-made objects, surface and internal ocean waves, rainfall effects on ocean returns, ocean currents, indirect studies of ocean-bottom topography, ship wakes, polar ice growth and migration, and glacial studies. Other uses include treaty verification and strategic reconnaissance [9]. An array of multifunction space-based radars is being considered for deployment in the near future for the latter purpose, and for the detection of airborne intruders and the protection of fleet battle groups [10]. One of the operating modes of such a network would be SAR imaging. Also, several sources reported that the launch of the space shuttle Atlantis in December 1988, carried a classified payload for the intelligence community which included a SAR [11], [12], [13].[1]

The number of spaceborne SARs is relatively small and includes both Earth-orbiting types and those which have been used in the exploration of the solar system. The first of the orbiting SARs was SEASAT, launched in 1978. In its short lifetime, this radar provided an abundance of data which are still being analyzed today. Of the series of Shuttle Imaging Radars, SIR-A and SIR-B have flown, with SIR-C and SIR-D being planned. The European Space Agency, Canada, and Japan are also planning Earth-orbiting SARs for the 1990s, including the Earth Observing System for monitoring global change, in conjunction with NASA [14]. The SARs which have left Earth's orbit include the lunar imaging experiment that was carried on Apollo 17 for mapping of the lunar surface and subsurface. Venus, with its perpetual cover of sulfuric acid clouds which are opaque to visible and infrared radiation, is an intriguing target for study by SAR [15]. In 1978, Pioneer Venus was sent there and mapped nearly all of the planet's surface with a resolution of 100 km. The Soviet Union in 1983 sent Venera 15 and Venera 16 to Venus and mapped about 30% of the surface, near the north pole, with a resolution of about 2 km. The launch of the Magellan (Venus Radar Mapper) craft, on May 4, 1989, will result in imagery with resolution of some 250 meters. Titan, a moon of Saturn, is also a candidate for study by SAR, since its surface is covered by clouds of methane. This moon is of particular interest since its atmosphere contains hydrocarbon molecules.

1.2 Types of Synthetic Aperture Radar

Synthetic aperture radars can usually be classified as either stripmap, spotlight, inverse, bistatic, or as a turntable imaging experiment. While these classifications are usually clear and unambiguous, some overlap is possible. Their descriptions follow.

[1] According to [13], the $500 million Lacrosse satellite "…will use a new type of radar imaging…." References [11] and [12] indicate more directly that the satellite contains a SAR. The satellite weighs 20 tons and has a 150 foot antenna, shown in a simulation picture as being oriented with its longest dimension perpendicular to the direction of travel.

Stripmap SAR is the oldest and most common type of SAR. The requisite motion between the radar and the object being imaged is provided by the platform carrying the radar, such motion usually being, at least approximately, a straight line. The radar's antenna points broadside (although squinting forward or backward is possible) and illuminates a strip on the ground with a succession of rapidly-transmitted pulses. Within the limits of the data storage medium and the time required to process the data collected, there is essentially no limit on the length of the strip imaged. With appropriate signal processing, cross-range resolution far better than the beamwidth of the antenna can be achieved—in fact, in what is called focused SAR, the resolution is independent of range.

In stripmap SAR and all other types of SAR, high range resolution is achieved by transmitting a wideband signal. Peak power limitations on the transmitter and the desire to maintain a useable signal-to-noise ratio dictate that the bandwidth cannot be achieved by simply transmitting a short pulse. Since the uncertainty principle from Fourier transform theory does not address the *maximum* bandwidth of a signal of a specified duration, it is possible to obtain the bandwidth required for adequate range resolution by transmitting a so-called sophisticated signal [16], [17]. Such signals can be of such a duration to attain sufficient energy on the target, yet have suitably large bandwidth. One such signal that has proved popular is the linearly frequency modulated signal, the instantaneous frequency of which varies linearly with time within the pulse. This signal provides an elegant solution to the problem, and hardware for both transmitting and receiving it can be built with relative ease. However, many other signals are possible (see [18], for example), and an increasing amount of digital hardware in the radar is making these more feasible. This dissertation does not dwell on the specific form of the transmitted signal, as explained in Chapter 3; the results obtained are more general than could be had by assuming a particular signal.

Spotlight SAR differs from stripmap SAR in that the antenna is pointed at a single patch on the ground as the radar passes by, so that the patch is eventually observed for a longer interval than would be possible with a fixed antenna. In an informal interpretation of Fourier transform principles, the longer the patch is observed, the more is known about it; radar engineers speak of increasing the coherent integration time. The added information is used to increase the resolution of the image of the patch which is reconstructed from the data. A substantial increase in resolution over stripmap mode techniques is possible. The disadvantage is that a smaller ground patch is imaged. Of course, it is possible to increase the coverage by forming a mosaic of spotlight-derived images, possibly by having multiple beams of a phased array, but then the disadvantage is seen as an increase of one in the order of the computational complexity. Nevertheless, the increased resolution available from spotlight SAR is often valued highly enough to override the disadvantages. Spotlight SAR is one of the types of radar which is the focus of this dissertation.

A third type of SAR is called inverse SAR [19]. While the appropriateness of the name may be questioned, the principle of operation derives from the fact that the radar is nominally fixed and

the relative motion is provided by the rotation of the imaged object. As pointed out in [20], inverse SAR and spotlight SAR are similar. The main difference, and not a minor one in implementation, is that inverse SAR must somehow discover the trajectory of the object in order for an image to be formed. The rotation must also be found or assumed to have a certain character during the imaging period. These tasks are substantial and prone to estimation errors, especially with non-cooperative targets, making inverse SAR more difficult than spotlight SAR and usually yielding lower-quality imagery. Applications of the method (not mentioned in the preceding section) include imaging a ship from its wave-induced motion, imaging (possibly maneuvering) aircraft, and identifying objects in Earth orbit [21], including satellites and "space junk." Early radar astronomy experiments in the early 1960s involved mapping the surface of the Moon from a radar on Earth [5], [22]. The new imaging methods developed in this dissertation will probably find less utility in inverse SAR than spotlight SAR for reasons having to do with the normally small angular extent of the imaged objects as seen from the radar. This is explained more fully in Section 1.3 below.

In a bistatic SAR, the transmitting and receiving antennas are separated. The motion relative to the target can be provided by either or both the transmitter and receiver. Bistatic SAR provides some advantages over monostatic SAR in certain situations. Among these is the possibility of a large, Earth-orbiting network of receivers with relatively few transmitters, for wide-area coverage. This arrangement could be more cost effective than equal coverage by a number of monostatic units. Another advantage is that since the receivers are quiet, they would be hard to locate, jam, or destroy. See [10] for more discussion of these ideas. Still another arrangement of the transmitter and receiver would have the transmitter on a satellite and the receiver, more vulnerable yet quiet, on an airplane near the ground patch being imaged. A disadvantage of bistatic SAR is the problem of maintaining coherence between the two units. This type of SAR is also a subject of this dissertation.

Another situation in which SAR is used as an imaging tool is that in which the object being imaged is placed on a turntable and illuminated from a fixed radar. This setup is usually associated with an experimental laboratory [5], [23], [24], and is used to test and develop new theories and algorithms. Although not conceptually different than spotlight SAR or inverse SAR, the controlled conditions allow certain "nuisance" factors to be virtually eliminated, such as the residual error in estimating the relative motion between radar and target that is usually removed using motion compensation and autofocus techniques, or problems with multiple-time-around echoes. Another significant difference from spotlight SAR, and frequently inverse SAR, is that with the target on a turntable, it is easier to get a much larger variation in look angle. While this is normally useful in increasing the resolution in the image, it can exacerbate the problems of motion through resolution cells and variable occlusion. For these reasons, this style of imaging is designated as a separate category. The work reported herein could have significance on imaging with such apparatus, since

space is often limited on radar ranges and the subtended angle of the target as seen by the radar can be relatively large.

Other SAR imaging modes are possible, or there may be variations which do not fit neatly into the categories above. For example, there could be a bistatic inverse SAR. Some of the work reported in [5] and in [24] used a bistatic configuration in conjunction with a turntable.

1.3 Problem Statement

Signal processing for SAR historically has been based on Fourier transform techniques. One reason for this is that when SAR was invented in the 1950s and for some time after that, the only way to process the huge amount of data that was collected in a reasonably expeditious manner was to use optical techniques, such processors being based, at least in part, on Fourier optical principles. With the advent of digital processing in recent years, the existence of efficient algorithms for the computation of the discrete Fourier transform has continued to offer compelling reasons to use Fourier-type reconstruction methods. Additionally, geometrically-related simplifications in most analyses have engendered the assumption of plane waves being present over the scene being imaged, again encouraging the use of Fourier techniques. In applications where the distance of the radar from the ground patch is very large compared to the size of the ground patch to be imaged, such processing is appropriate, though still an approximation. In other cases, the wavefront curvature cannot be ignored and other steps must be taken in order to yield high-quality imagery.

The geometric assumption that is almost universally used in SAR will now be examined, with reference to Fig. 1. 1. The ground patch is a disk of radius L centered on an x-y coordinate system. The radar, for now, is at $(-R, 0)$. The wavefront, which is defined here as the locus of equal times-of-flight (for a photon of a given pulse, round trip), is a circle (sphere) of radius $r = R + x_0$ passing through the point (x_0, y_0). Points lying very near to this locus will reflect energy back to the radar with approximately the same delay. If $R \gg L$, then

$$\sqrt{1 + \frac{y_0^2}{(R + x_0)^2}} \approx 1$$

and

$$(R + x_0)\sqrt{1 + \frac{y_0^2}{(R + x_0)^2}}$$

is a region of such points. However,

$$(R + x_0)\sqrt{1 + \frac{y_0^2}{(R + x_0)^2}} = \sqrt{(R + x_0)^2 + y_0^2}\ ,$$

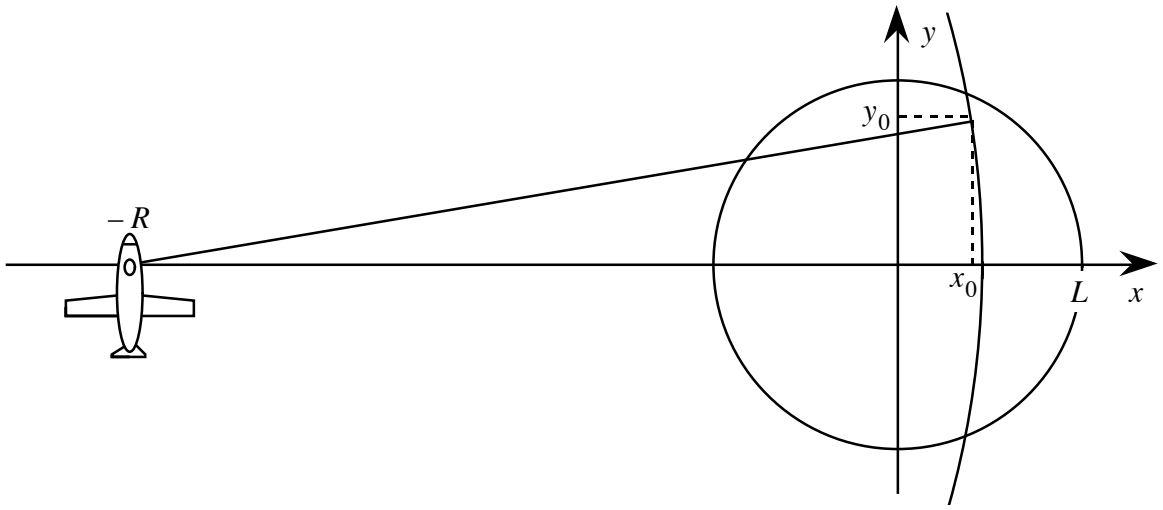


Fig. 1. 1. Geometry for showing the plane wave assumption which is common in the analysis of synthetic aperture radar.

the right-hand side of which is the locus of equal times-of-flight for a plane wave—the vertical line through (x_0, y_0). Although other things need to be considered in determining when this approximation is acceptable, the basic assumption is that as long as $R \gg L$, the plane wave approximation is valid. It should be noted that the definition of wavefront, above, is a far-field definition in that differences in times-of-flight from different parts of the antenna are assumed negligible.

The connection between plane waves and the Fourier transform is intimate. The two-dimensional Fourier transform of a function $g(x, y)$,

$$G\left(k_x, k_y\right) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \exp\left[-j\left(k_x x + k_y y\right)\right] dy\, dx \ ,$$

is seen to be a decomposition of $g(x, y)$ into plane waves of all orientations and wavelengths. Diffraction studies use the Fourier transform as a fundamental tool. Therefore, given that most SAR analyses have assumed that plane waves are present over the scene to be imaged, it is only natural for traditional image reconstruction methods to be based on the Fourier transform and its inverse. These points will be expanded in the next chapter in a discussion of the state of the art in SAR image reconstruction.

In some applications, it may be desirable or necessary to expand the area that is imaged or to reduce the distance from the radar to the center of the ground patch, or both, to such an extent that the relationship $R \gg L$ is not valid. If Fourier inversion methods, which are based on the plane wave assumption, are then used, the image quality will suffer. Specifically, the image will show defocusing artifacts, or loss of resolution, at points increasingly removed from the image center.

Such applications include the nighttime and poor-weather surveillance of the U.S.S.R. in which coverage of huge expanses with high resolution is required for the purposes of arms treaty verification and the monitoring of other, legal, activities of a strategic nature such as the movement of SS-24 and SS-25 mobile missiles [9]. Most other satellite-based Earth-observing SARs are also candidates for such improvements. Another application is battlefield surveillance in which a helicopter or remotely piloted vehicle pops up for a quick look behind enemy lines [6], [7]. In applications such as these, it is common to discard a good deal of the available data through the process of presumming [25], i.e., a low-pass filtering operation in which the Doppler bandwidth is effectively reduced. This also effectively reduces the beamwidth of the antenna so that the illuminated spot need not be defined by the actual antenna beam. There are at least four reasons that this is done. The first is that it may not be necessary to image a larger area. The second is that computational power may be limited, and imaging the smaller area is a compromise in that it requires less computation. The third may be that a suitable reconstruction algorithm may be lacking to handle the larger area. The fourth reason is that system design constraints may not allow the antenna to be made larger to narrow its beamwidth. Whatever the reasons have been in the past, the applications cited eliminate the first one and faster computers and new architectures are tending to reduce the second reason. This dissertation should be a step toward eliminating the third reason. Finally, as mentioned earlier, another application which could benefit is the laboratory-turntable experiments in which the test range may be smaller than optimum.

Therefore, the problems addressed by this dissertation are to define new SAR imaging modes which arise when the plane wave assumption is relaxed and to derive and simulate new image reconstruction algorithms for some of them. The important observation in [26] relating the conceptual similarity of SAR and computer-aided tomography (CAT), an imaging technique of medicine which uses collimated X-rays, provides a basis for the new ideas presented herein.

1.4 Organization and Summary

Since this work borrows from both the SAR and the CAT fields, it is necessary to be cognizant of both in order to understand the new results. Chapter 2 provides a brief introduction. The discussion is sometimes somewhat broader than necessary in order to allow some speculation on unsolved problems or alternate solutions to the problems considered in this dissertation. The major assumptions which are made in the SAR case are identified and a brief mention is made of the various interpretations used in understanding SAR, of which the tomographic interpretation is apparently the most recent. Related imaging techniques are listed, but no attempt is made here to compare or to unify them, or to adapt them to the SAR problem.

Chapter 3 describes the methodology used in the later simulations, including special adjustments which had to be made in the absence of the rich Fourier theory that accompanies plane wave studies. Details which affect most or all of the simulations are specified; an attempt was made to

keep these details constant throughout, as much as possible, so that the various reconstructions can be compared. A test function is chosen, and a set of baseline reconstructions using a conventional CAT algorithm (known as convolution-backprojection) and using data that would result from plane waves interacting with the ground patch is made for comparison with reconstructions in later chapters which use new methods.

Chapter 4 deals with the problem of correcting for the effects of wavefront curvature in monostatic spotlight SAR [27]. The wavefront curvature is motivated using the plane wave spectrum, or angular spectrum, concept. This is explicitly stated partly to defuse a common misconception that the far field of an antenna might be a plane wave. Here, as in the other simulations, it is necessary (or at least highly desirable) to find the *projections*, which are analogous to the X-ray data of CAT, in closed form. A basis for understanding the new algorithm is given and computational requirements are considered. Various reconstructions are displayed for several trajectories of the radar around the ground patch, all of which exhibit extreme amounts of wavefront curvature. The algorithm is shown to provide reconstructions of a quality similar to the baseline reconstructions of Chapter 3. The chapter closes with comments on how the new algorithm affects aspects of SAR known as "quadratic phase."

Chapter 5 reports similar results as Chapter 4, only for bistatic spotlight SAR [28], [29]. In bistatic SAR, the correction for attenuation of the outgoing and returning waves greatly complicates the reconstruction process. Two algorithms are described, one of which makes the correction and a simpler one which should be useful in those instances in which the attenuation can be ignored. While the spherical-wave monostatic case required the introduction of a new variable, R, the bistatic case requires two such variables, one each for the transmitter and the receiver, plus a "program" for relating the positions of the two units during data collection. The algorithms yield reconstructions which are similar in quality to the baseline reconstructions.

Chapter 6 introduces two additional problems which appear to have solutions by tomographic methods, but for which actual algorithms yet have to be developed. The first of these is the correction for nonuniform illumination of the ground patch due to the pattern of the radar antenna. With the solution of the wavefront curvature problem in hand (Chapters 3 and 4) and the ability to image larger scenes, this problem becomes more important. A byproduct of this investigation is a duality theorem which is a generalization of the well-known Projection-Slice Theorem. The second problem proposed is that of imaging a rapidly-spinning object, for example in inverse SAR mode, or imaging a stationary object from a rapidly moving platform. This problem may be important and evidence is offered suggesting that ultra-high resolution imaging from satellites may be a candidate for more sophisticated signal processing. In particular, an unusual kind of fan-beam projection is identified, one which approximates the fan beams of modern CAT scanners under a reasonable range of parameters. Encouraged by the success achieved by applying tomographic reconstruction

concepts so far, Chapter 6 continues by proposing a set of problems which would appear to yield to more efforts of a similar nature.

Appendices A, B, and C contain details of subsidiary points. Appendix D contains listings of some of the computer programs that were used in the dissertation.

It may be helpful for the reader to know what the author believes to be new, as represented by this dissertation. A few brief points will be made in that regard. The central focus of the work is to examine the algorithmic implications of nontrivial amounts of wavefront curvature in spotlight SAR. There has been almost no work on this problem reported in the literature. Of the references cited here, only two make mention of the problem; neither tries to repair it. Therefore, the clear statement of the problem may be new, and certainly all of the algorithms are new. The methodology that was developed, as described in Chapter 3, may be a worthwhile contribution in that it extracts the essence of the problem while suppressing the more traditional radar issues which have been studied before by others. Included in this methodology is the calculation of several types of projections of a useful test function, all in closed form. Finally, the problems and imaging modes mentioned in Chapter 6 are believed to be new.

CHAPTER 2
BACKGROUND

SYNTHETIC aperture radar and computer-aided tomography are examples of a general class of problems known as linear inverse problems with discrete data [30]. In signal processing language, linear inverse problems can be stated as, "Given that a system is linear (and possibly time-varying) and given an input and an output, find the system" — in a broad sense, deconvolution. In the language of mathematical physics, the mathematical formulation produces an integral equation of the first kind [31]. Despite this commonality, SAR and CAT have historically been studied by nearly disjoint groups of people. Even though they fall into the same class of problems, there seemed to be little benefit to be had from studying both types of imaging. It was in the early 1980s, most importantly with the work described in [26], that a closer connection was made between SAR and CAT. Specifically, that connection is the observation that SAR is a bandlimited version of CAT. With this observation, it is possible to apply suitably-modified versions of algorithms which have enjoyed wide use and success in CAT to the problem of inverting SAR data to reconstruct images of the magnitude of the complex reflectivity of ground patches or other scenes [32].

This chapter will serve as a brief introduction to these two imaging techniques.

2.1 Computer-Aided Tomography

Tomographic concepts are usually thought of as belonging to the field of medical imaging, but these principles have been observed, often independently, by researchers in a variety of disciplines. The mathematical ideas were first formalized by Johann Radon in 1917 [33]. (An English translation of Radon's paper appears in [22].) Besides various medical imaging modes, some of the application areas are radio astronomy, planetary occultation studies, statistics, partial differential equations, geophysics, optics, electron microscopy, nuclear magnetic resonance, nondestructive testing, remote air-pollution monitoring, chromosome studies, and others [22]. Here, the X-ray imaging methods of CAT will serve as the basis for discussion. [2] Several books and papers offer excellent

[2] Tomography comes from the Greek word *tomos* meaning slice or section. The word is appropriate for CAT since the human body is usually imaged in narrow cross sections, the cross sections being stacked when a three-dimensional image is needed. The usage seems tenuous with SAR and several other applications since the concept of imaging in discrete slices is absent. There is a possible connection of the word to the Projection-Slice Theorem, but, as seen in Chapter 4, even this is absent in the work of this dissertation. However, the word will be used here to mean generally the concepts associated with the Radon transform and its inversion, or any of the substantial modifications which are introduced here.

coverage of the subject [22], [34], [35], [36], [37], [38], [39], [40], [41].

X-rays experience an attenuation when passing through a material object. This phenomenon can be used to ascertain the internal structure of the object. First, consider a monochromatic beam of X-rays with intensity I_0 incident on a uniform slab with absorption parameter ρ_0 and thickness l_0. The intensity of the output beam is

$$I_1 = I_0 \exp(-\rho_0 l_0).$$

If a second uniform slab with characteristics ρ_1 and l_1 is abutted to the first, then the output intensity is

$$I_2 = I_1 \exp(-\rho_1 l_1) = I_0 \exp(-\rho_0 l_0) \exp(-\rho_1 l_1).$$

(Reflected energy at the interface complicates the situation, but is not important for the present discussion.) If N such slabs are involved, the output intensity is

$$I = I_0 \exp\left(-\sum_{i=0}^{N-1} \rho_i l_i\right).$$

In the case of a material with continuously-varying absorption, the above can be expressed as

$$I = I_0 \exp\left(-\int_{-\infty}^{\infty} \rho \, dl\right).$$

Now consider a two-dimensional distribution of X-ray-attenuating matter with absorption $f(x, y)$. As shown in Fig. 2. 1, an X-ray emitter and a detector are on rotating machinery such that an idealized beam of zero width passes through the object $f(x, y)$ and is measured upon exiting. Let points in the rotated coordinate system be denoted (x', y'), x' being the distance of the emitter-detector from the y' axis and the beam being parallel to the y' axis. The rotation angle is θ. If the incident intensity is $I_0(x')$, then the detected intensity is

$$I(x', \theta) = I_0(x') \exp\left[-\int_{-\infty}^{\infty} f(x, y)\, dy'\right].$$

Let $x' = p$ and define the *straight-line projection* or *Radon transform*

$$f_l(p, \theta) = -\log \frac{I(x', \theta)}{I_0(x')} = \int_{-\infty}^{\infty} f(x, y)\, dy'. \qquad (2.\ 1)$$

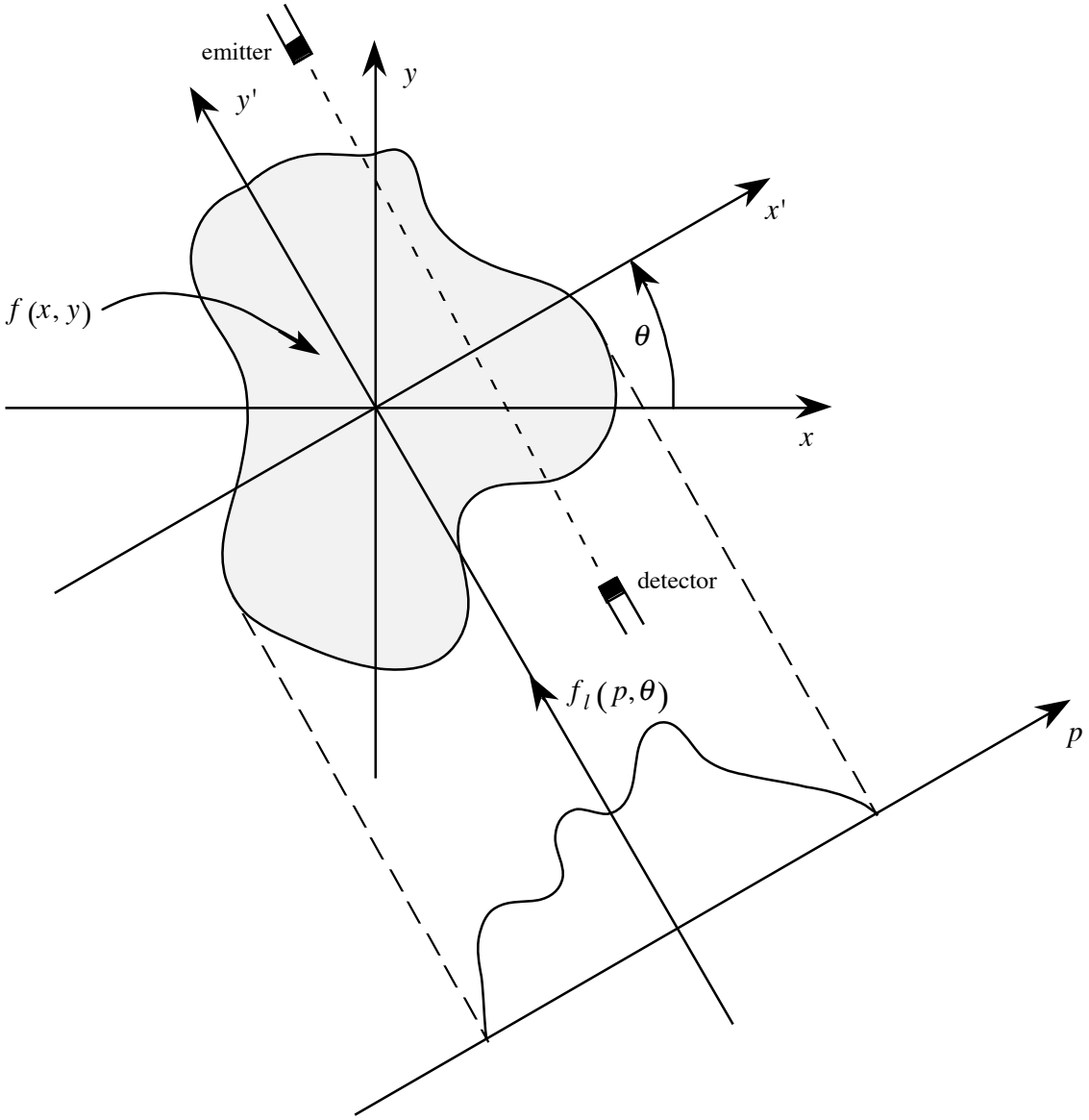


Fig. 2. 1. Geometry of the rotating emitter-detector machinery for data collection in X-ray computer-aided tomography.

Note the symmetry relation

$$f_l(p, \theta) = f_l(-p, \theta + \pi) = f_l(p, \theta + 2\pi), \tag{2. 2}$$

the first of which implies that, at least in the absence of noise, the projections need to be collected only over a range of 180°. In the above, the variables p and θ are not to be taken as polar coordinates [22], [34] because $f_l(0,\theta_1) \neq f_l(0,\theta_2)$ for $\theta_1 \neq \theta_2$, which is a pathological condition in polar coordinates.

The projection-slice theorem is not only an important conceptual tool in CAT but is also of great help in finding certain types of reconstruction algorithms. To derive this important result, it is helpful to first put (2. 1) into a different form. Define the vector

$$\mathbf{x} = (x, y)$$

and the unit vector in the direction θ

$$\xi = (\cos\,\theta, \sin\,\theta).$$

The notation $f_l(p, \theta)$ and $f_l(p, \xi)$ will be used interchangeably. The line integral on the right-hand side of (2. 1) can be written more clearly as

$$f_l(p, \xi) = \iint f(\mathbf{x})\,\delta(p - \xi \bullet \mathbf{x})\,d\mathbf{x} \tag{2. 3}$$

where the locus $p = \xi \bullet \mathbf{x}$ is a line a distance p from the origin and orthogonal to the unit vector ξ. All integration limits will be from $-\infty$ to $+\infty$ unless stated otherwise. Let $\mathsf{F}_p\{f_l(p, \xi)\}$ denote the Fourier transform of $f_l(p, \xi)$ with respect to p. Then

$$\mathsf{F}_p\{f_l(p, \xi)\} = \int f_l(p, \xi)\exp(-j\omega p)\,dp$$

$$= \int\left[\iint f(\mathbf{x})\,\delta(p - \xi \bullet \mathbf{x})\,d\mathbf{x}\right]\exp(-j\omega p)\,dp$$

$$= \iint\left[\int f(\mathbf{x})\,\delta(p - \xi \bullet \mathbf{x})\exp(-j\omega p)\,dp\right]d\mathbf{x}$$

$$= \iint f(\mathbf{x})\exp(-j\omega\,\xi \bullet \mathbf{x})\,d\mathbf{x}$$

$$= \iint f(x, y)\exp[-j\omega(x\cos\,\theta + y\sin\,\theta)]\,dx\,dy$$

With Fourier-domain variables Ω_1 and Ω_2 and the polar coordinate

$$\omega = \left(\Omega_1^2 + \Omega_2^2\right)^{1/2}$$

the desired result can be stated as

$$\mathsf{F}_p\{f_l(p, \xi)\} = \mathsf{F}_{x,y}\{f(x, y)\}\Big|_{\substack{\Omega_1 = \omega\cos\theta \\ \Omega_2 = \omega\sin\theta}}$$

where the notation on the right-hand side means two-dimensional Fourier transformation. In words, the one-dimensional Fourier transform of the projection of f at an angle θ is a slice of the two-dimensional Fourier transform of f through the origin at an angle θ.

Two incidental comments will be made about the projection-slice theorem. First, the main result is commonly used by antenna engineers in computing various "cuts" of the far-field radiation pattern of two-dimensional radiators. However, it is suspected that it is not usually known that there is a body of underlying theory that can be put to use in that application. Second, the effect of having a beam of non-zero width is to low-pass filter the projections. For example, if the beam is uniform, then the measured projection is the true projection convolved with a rectangular pulse. The effect on the two-dimensional Fourier domain data is a circularly-symmetric low-pass operation.

The projection-slice theorem immediately suggests one way to reconstruct an image from its projections: the so-called direct Fourier inversion [42], [43] (see also most of the general references on CAT). If the Fourier transform of every projection of f is known, then the two-dimensional Fourier transform of f is determined. In practice, only a finite number of the projections are known in θ, and they in turn are usually sampled in p. If the discrete Fourier transform (DFT) of each sampled projection is computed, then (assuming no aliasing problems are present) the two-dimensional Fourier transform of f is known on the polar grid shown in Fig. 2. 2. The inverse transform can be computed directly from these points, or, in order to take advantage of the efficiency of any of the fast Fourier transform algorithms, the data can be interpolated onto a regular Cartesian grid such as the one shown overlaid in solid line on the polar grid in Fig. 2. 2. The interpolation problem has been studied extensively [44], [45].

Another method of reconstructing an image from its projections is suggested by the projection-slice theorem, although it is not as obvious as the above method. If the Fourier transform of $f(x, y)$ is denoted by $F(\Omega_1, \Omega_2)$, then the inverse Fourier transform is written

$$f(x, y) = \frac{1}{4\pi^2} \iint F(\Omega_1, \Omega_2) \exp(j\Omega_1 x + j\Omega_2 y)\, d\Omega_1\, d\Omega_2 .$$

Converting to the Fourier-plane polar coordinates (ω, θ), this can be written as

$$f(x, y) = \frac{1}{4\pi^2} \int_0^{\pi} \int_{-\infty}^{\infty} F(\omega\cos\theta, \omega\sin\theta) \exp[j\omega(x\cos\theta + y\sin\theta)] |\omega|\, d\omega\, d\theta \qquad (2.4)$$

or, by using the projection-slice theorem and the conversion to the rotated system

$$p = x\cos\theta + y\sin\theta,$$

the equivalent form

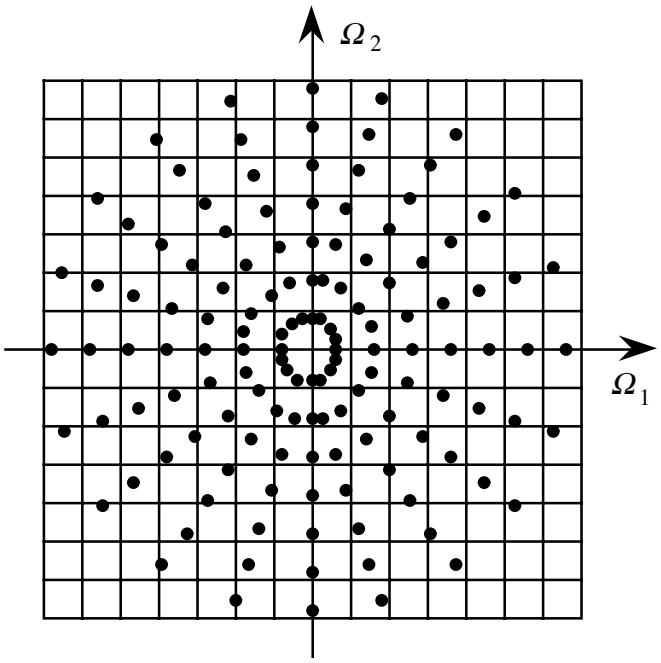


Fig. 2. 2. Polar grid showing the locations in the Fourier plane at which the two-dimensional Fourier transform of the object function is known, for direct Fourier inversion. Also shown is a rectangular grid to which the polar data can be interpolated for more efficient inversion.

$$f(x, y) = \frac{1}{4\pi^2} \int_0^{\pi} \int_{-\infty}^{\infty} \mathsf{F}_p\{f_l(p, \xi)\} \exp(j\omega p) |\omega| \, d\omega \, d\theta$$

can be derived. The inner integral is the inverse Fourier transform with respect to ω of the product of the Fourier transform of a projection with $|\omega|$. This filtering operation can be expressed in the spatial domain as the derivative of the Hilbert transform of the projection. Defining this filtered version of the projection as

$$\begin{aligned} \bar{f}_l(p, \xi) &= \frac{d}{dp} \int \frac{f_l(t, \xi)}{p - t} \, dt \\ &= f_l(p, \xi) * k(p), \end{aligned} \tag{2. 5}$$

the outer integral becomes

$$f(x, y) = \frac{1}{2\pi} \int_0^{\pi} \bar{f}_l(p, \xi) \, d\theta \tag{2. 6}$$

which is called backprojection. This is a slightly subtle operation which is perhaps better understood in a discrete version, such as would be used in an actual computer implementation. In this case, the integral is replaced by a summation over a finite set of filtered projections. For each $\bar{f}_l(p, \xi)$, a new, two-dimensional function is formed which is equal to $\bar{f}_l(p, \xi)$ along lines parallel to the horizontal axis and which is constant along lines parallel to the vertical axis. This new function is then rotated by the correct θ, the angle from which it was originally projected. All such functions are then added and the total is divided by 2π. An example of a single back-projected filtered function will be shown in Chapter 3—a picture makes the above description superfluous (see Fig. 3. 5). This reconstruction process is called *convolution back-projection*, or sometimes *filtered back-projection.* Since the algorithm is described completely in the spatial domain, [3] without using the Fourier domain, it is known as a spatial-domain reconstruction algorithm, even though in this derivation the projection-slice theorem was used in an intermediate step.

When implementing convolution back-projection in a computer program, there are a number of details which need to be filled in, including setting up data structures and several numerical details. One of the numerical details which needs attention in any implementation is a one-dimensional interpolation between the discrete points in the filtered projections and pixel locations in the reconstructed-image plane. Another issue which is not covered here is that of adequate sampling in both p and θ—see [38] and [46] for more on this. These and some other details will be elaborated upon in Chapter 3 and chapters following, as appropriate. In any case, the final authority on what was actually done to reconstruct the various images which follow is the collection of computer programs included in Appendix D.

Other reconstruction methods which differ markedly from the two described here are also used. These include the algebraic reconstruction techniques and related methods [34], and the circular harmonic decomposition [22], [47], [48], [49]. None of these methods are developed for SAR in this dissertation; however, algebraic reconstruction methods can clearly be adapted for unusual projectional geometries such as exist in the wavefront curvature problem in SAR, if the basis functions are chosen to be individual pixels in the reconstruction plane. Pertinent work which sheds new light on the convolution back-projection method and unites several other methods under a common framework is discussed in [50], [51], [52]. The Radon transform also bears some resemblance to the Hough transform ([22], [53], [54], [55]).

2.2 Synthetic Aperture Radar

Synthetic aperture radar can be understood in more than one way. The two interpretations that are traditional are those of processing Doppler shifts and a sequentially-realized array antenna (a synthetic aperture). Two interpretations that are less commonly used are the tomographic view-

[3] This is the common interpretation. Perhaps it is better to consider the projection data as existing in the Radon transform domain, much as the Fourier transform domain is used in other situations.

point, as elucidated in [26], and the matched filter. This section will briefly discuss the first two and the last of these; the tomographic interpretation will be explained more fully since it is the crux of this dissertation. The discussion will focus on spotlight-mode SAR. (See [56] or [57] for excellent discussions and interpretations of strip-map SAR.) First, some common assumptions which are made in SAR will be stated. For expository purposes, the plane-wave assumption will be used in this chapter.

Several references contain information on SAR and related subjects, in addition to those already mentioned: [58], [59], [60], [61], [62], [63], [64].

2.2.1 Assumptions

Several "standard" assumptions are commonly made in SAR work. Generally speaking, these apply here as well, except for the aforementioned discarding of the plane-wave assumption. The first of these is that the scene being imaged reflects electromagnetic waves with an attenuation in magnitude and a shift in phase. These two quantities are lumped together into a single complex-valued reflection coefficient; it is the magnitude which is ordinarily to be determined by the imaging process. This reflection coefficient is assumed to be constant over the frequency band that is interrogated by the radar. It is also assumed to be independent of the incidence angle of the interrogating signal and the angle at which the reflections are measured. This further implies the assumption that variation in occlusion effects are negligible with respect to angles. Another assumption concerning the manner in which the electromagnetic energy interacts with the object is that secondary and higher-order reflections have a significant effect on the total returned signal only within a region that is no larger than a resolution cell. These reflections sum in phasor fashion and are integrated into a single reflection coefficient that characterizes the entire cell. If multiply-reflected energy does find its way farther than one resolution cell from its initial point of incidence, it is assumed to be small enough to have negligible effect on the quality of the reconstructed image. It is conjectured here that the reason that such a coarse treatment of diffraction is acceptable is because radar imaging rarely, if ever, strives to approach the wavelength limit in resolution. Perhaps as SAR resolution increases, the SAR engineer will have to develop new algorithms which account more accurately for diffraction, just as acoustic tomographers do now.

Another statement of the scattering process is that if an impulse is transmitted from the radar, then at some later time it reflects from any scatterer at a particular distance from the transmitter. A portion of the incident energy (as determined by the reflection coefficient for each scatterer) is reflected in the direction of the radar and collected at a later instant. This scattering model appears to require some rectification with the earlier model—perhaps such could be had by assuming two different reflectivity functions, one in which the net reflection properties of each resolution cell are lumped into a single scatterer at the center of the cell, for each look angle. A perusal of the radar cross section literature would be appropriate for the interested person, with attention to such details

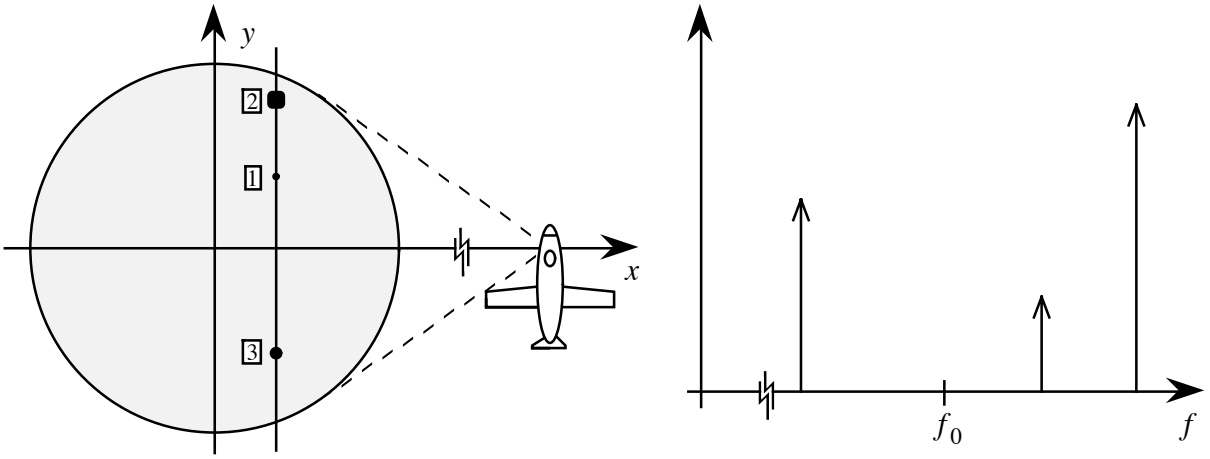


Fig. 2. 3. (a) Geometry for the Doppler interpretation of SAR showing three point scatterers of different strengths. (b) Approximate spectrum for the three point scatterers of (a).

as the Born approximation and the physical optics approximation.

Another assumption which falls into the category of diffraction effects is that of linearity.

As discussed earlier and as will be elaborated later in a discussion of the plane wave spectrum, all points of the ground patch are assumed to be in the far field of the antenna (both antennas, in the case of bistatic SAR). No restriction is placed on the size of the ground patch.

Two more assumptions used here are that the ground patch is flat and that the radar is carried at zero altitude. The algorithms presented later are believed to be adaptable to more general geometries when these assumptions prove to be troublesome.

One final assumption that holds at all times except for a peculiar situation described in Chapter 6 is that the radar is stationary while transmitting and receiving, and that the ground patch is also stationary while the pulse is traveling over it. This "stop-and-go" model seldom needs stating because the effects are often very small; however, there is some evidence that with the platform velocities involved with imaging from space, there may be some deleterious effects on image quality if this assumption is retained

2.2.2 Doppler interpretation

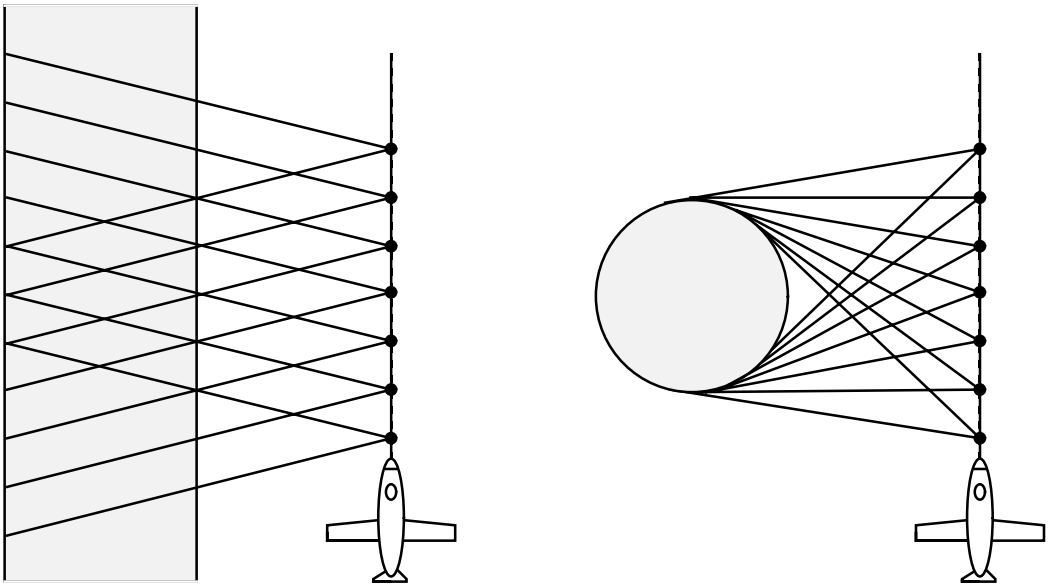
The Doppler interpretation of spotlight SAR will be explained with the aid of Fig. 2. 3 (a). For a somewhat simplified discussion, assume that the radar transmits a sine wave of frequency f_0 and moves continuously around the ground patch, which is at a great enough distance that plane waves are present over it. Notice that the "stop-and-go" assumption is abandoned, since it does not make sense here. It makes no difference here whether the radar is said to move around the ground patch or whether the radar is fixed and the ground patch rotates, in turntable fashion. There

are only three point scatterers in a line parallel to the y axis. The variation in look angle is small and the radar is near the x axis, illuminating the entire ground patch. The sine wave is reflected from scatterer 1 and experiences a Doppler shift [4] which is proportional to its closing velocity and therefore its distance from the origin, and has a magnitude which is proportional to the reflection coefficient of that scatterer. The same happens for scatterer 2, here shown to be a stronger reflector. Scatterer 3 induces a negative Doppler shift. The spectrum of the returned signal is shown in Fig. 2. 3 (b). Downconversion by f_0 of that signal to baseband followed by Fourier transformation and display would give an image of the three scatterers that has resolution far better than that attainable with the broad antenna beam.

In order to achieve range resolution, the radar must transmit pulses, not a sine wave. (All scatterers on a line passing through scatterer 1, for example, and parallel to the x axis will induce the same amount of Doppler shift. This fact is exploited in Chapter 6, Section 3.) If short pulses are transmitted, or if a pulse compression system is employed, the effect is to measure samples of the Doppler-shifted baseband signals, even with stop-and-go motion of the radar. To avoid serious degradation to the image, ordinary practice concerning Nyquist sampling of the Doppler signals should be observed by adjusting the pulse repetition frequency (PRF). If the antenna illuminates an area larger than that which is to be imaged, the PRF needs to be high enough to avoid aliasing the returns from scatterers at the edge of the antenna beam. To reduce the data rate before image reconstruction, the sampled data can be low-pass filtered and resampled at a lower rate; this is called presumming [25].

Problems can appear if the total rotation of the patch during the above coherent processing interval is too large, however. At first, the achievable resolution increases with increased look angle, since the data record before Fourier transformation is longer. Eventually, though, the Doppler shift from a particular scatterer will change, as its distance from the x axis changes, and its Doppler spectrum will begin to broaden, causing defocusing. The change in instantaneous Doppler can become so great that the scatterer can show up in more than one Doppler cell, such as would be found in a signal processing apparatus which computes discrete frequency bins to form the image, e.g., the discrete Fourier transform. Similarly, the change in range to the scatterer can become so great that it will appear in more than one range cell. These and other problems are discussed extensively in the literature and show up a weakness in simple Doppler-based signal processing. Yet, this style of thinking is very useful in understanding the image formation process, at least intuitively. An elegant way of handling this problem is through polar formatting [26], [65], [66]. The tomographic methods to be discussed later are examples of polar formatting.

[4] The model used here for induced Doppler shifts assumes a narrowband signal (i.e., small percentage bandwidth), and is exact only for sine waves. The Doppler process (ignoring relativistic effects) is fundamentally a time-scaling of the shifted signal, or a complementary inverse scaling of the frequency axis of its Fourier transform. For narrowband signals, this is approximated by a simple frequency shift. Interestingly, the ambiguity function, a common tool in radar signal analysis, assumes that the Doppler-affected signal are shifted, not scaled. The fact that the ambiguity function requires narrow band signals may not be generally appreciated.



(a) (b)

Fig. 2. 4. (a) Geometry for stripmap SAR. (b) Geometry for spotlight SAR. Distances between the aircraft and ground patch are shown reduced.

2.2.3 Synthetic aperture interpretation

The second traditional interpretation of SAR is the one which gives it its name: the formation of a "synthetic" or sequential aperture or array. Using only the real antenna mounted on the radar platform, crossrange resolution varies with range and is given by $\delta_{cr} = \theta_b R$. Increasing the size of the antenna's crossrange dimension decreases its beamwidth θ_b, but increasing it to the size needed for a useful δ_{cr} is not generally practical by a huge margin.

Stripmap SAR is the classic SAR imaging mode, and it is in this context that the synthetic aperture concept is almost always used. It is instructive to look at this case before examining the spotlight mode. The appropriate geometry is shown in Fig. 2. 4 (a). The radar normally travels a nominally straight path with its antenna pointing broadside. The return signals are recorded and processed coherently so that the net effect is that of a very long array with copies of the real antenna at each element position. Without correction, the array is focused at infinity. Frequently, phase adjustments are made to each return so that it is focused somewhere on the strip, or even at each range of interest. When focused in this manner, the resolution is independent of range, and under a reasonable range of approximations it is equal to half the length of the real antenna.

The above description of stripmap SAR needs one correction. The effect is not the same as an ordinary array of the same long length which transmits and receives simultaneously on all elements. In such an array, a signal which leaves a particular element would later be received on *all* the other elements — a matrix whose i–jth entry is the transmission from the ith array element to

the *j*th element would in general have no zeros in it. For the synthetic array, there is no transmission from the *i*th element to the *j*th element if $i \neq j$ — the corresponding matrix would be diagonal. Accordingly, standard array analysis should be applied with care. This is presumably the source of the difference in the SAR two-way pattern and the two-way pattern of a real array as given in [67].

The achievable resolution of stripmap SAR is limited by the distance between the position of the antenna when a point is first illuminated and its position on the final illumination, that is, the effective synthetic array length; spotlight SAR overcomes this limitation by steering the antenna so that the dwell time is increased, as shown in Fig. 2. 4 (b). Again, focusing can be applied to achieve the effect of the radar traveling a circular trajectory centered on the ground patch. The discussion in the previous paragraph about simultaneous versus sequential arrays applies here. In addition, standard array analysis falls short here because since the antenna is steered, the composite pattern cannot be separated into an array factor and an element factor in the usual way, the underlying spatial convolution having been altered. Nevertheless, the synthetic array concept holds even if nonstandard methods are required to analyze it. The achievable cross-range resolution, given in [26], is

$$\delta_{cr} = \frac{\pi c}{2\omega_0 \sin \theta_M}$$

where c is the speed of light, ω_0 is the center frequency in radians per second, and θ_M is half the variation in look angle. This holds for a reasonable set of approximations, including $\theta_M \ll 1$. Cross-range resolution is not determined by the effective array length as in stripmap SAR.

It is important to note that the Doppler concept is an artifice, if a useful one. As is apparent in the synthetic aperture interpretation, it makes no difference what path the radar takes between pulses; the only thing that matters is the vector displacement. Also, it is important to distinguish between interpulse Doppler and intrapulse Doppler. Since the "stop-and-go" model is almost universally used, and since it is possible to get images even if that were the actual data collection style, it is obvious that interpulse Doppler is the usually desirable quantity. In fact, the presence of large amounts of intrapulse Doppler can lead to image quality degradation. This will be discussed further in Chapter 6.

2.2.4 Tomographic interpretation

The tomographic interpretation is the most important one for this dissertation because it allows the convolution-backprojection algorithm to be generalized to account for wavefront curvature and other effects. The development here is in the manner of that in [26], but differs slightly and stops short of specifying a form for the transmitted signal. With reference to Fig. 2. 5, the distance to the radar from the origin is R and the radius of the ground patch $g(x, y)$ is L. The reflectivity is assumed to be zero outside that circle, either as a result of limiting by the antenna beam, pre-

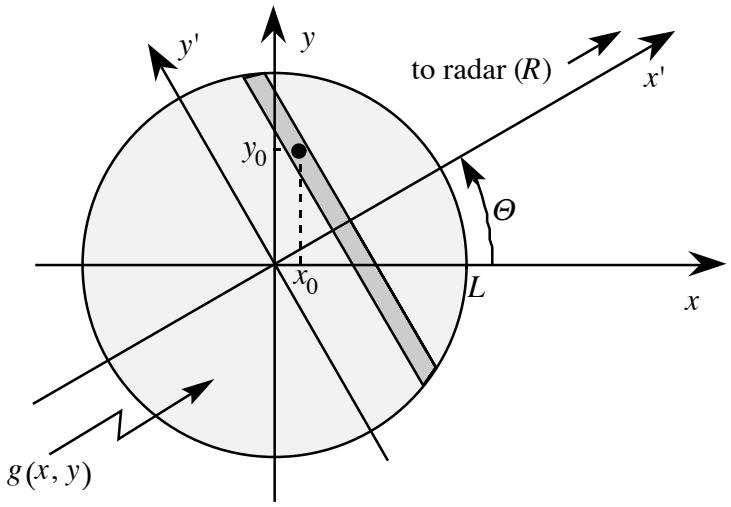


Fig. 2. 5. Geometry for deriving the return signal and showing the similarity to CAT.

summing, or other means. Assume $R \gg L$. The radar transmits a signal $s(t)$ which reflects from a differential area scatterer at coordinates (x_0, y_0), or (x'_0, y'_0) in the Θ-rotated system.[5] Let the round-trip attenuation be A. Then the returned differential signal is

$$r_0(t) = A\, g(x,y)\, s\left[t - \frac{2\,(R - x'_0)}{c}\right] dx'\, dy'.$$

With $R \gg L$, the return from a line of scatterers parallel to the y' axis is

$$r_1(t) = \int A\, g(x,y)\, s\left[t - \frac{2\,(R - x')}{c}\right] dy'\, dx'.$$

The attenuation A can be considered constant over the ground patch, so the above reduces to

$$r_1(t) = A\, s\left[t - \frac{2\,(R - x')}{c}\right] \int g\,(x,y)\, dy'\, dx'.$$

But the integral above is the projection defined on the right-hand side of (2. 1) or (2. 3):

$$\int g(x,y)\, dy' \;=\; \int g\,(x,y)\, \delta(p - \xi \bullet \mathbf{x})\, d\mathbf{x} \;=\; g_l(x', \Theta)$$

so that

[5] Radar coordinates will be in upper-case letters.

$$r_1(t) = A\, s\left[t - \frac{2\,(R - x')}{c}\right] g_l(x', \Theta)\, dx'.$$

The total return is obtained by integrating over x':

$$r(t) = A \int s\left[t - \frac{2\,(R - x')}{c}\right] g_l(x', \Theta)\, dx'.$$

A short series of standard abuses in notation allows this to be written as a convolution. First, the time origin is shifted by $2R/c$, then the unit of measure for distance is adjusted so that $c = 2$. Let the propagation attenuation be ignored, since it is considered a constant here. (This assumption will be eliminated later.) Adding a subscript to indicate the direction of the radar for the particular pulse, the result is

$$r_\Theta(t) = \int s\,(t + x')\, g_l(x', \Theta)\, dx' = s(t) * g_l(-t', \Theta). \tag{2. 7}$$

In other words, the return signal is the transmitted signal convolved with a reversed version of a projection of the ground patch. One can envision the radar pulse travelling over the ground patch physically implementing the convolution. Note that with $c = 2$, it does not matter mathematically whether the convolution is in time or space. The sign which is added in order to reverse one of the functions that is involved in the convolution could be eliminated (see [26]) by placing the radar on the negative x'-axis. The present method is used because it simplifies the simulations in later chapters.

The relationship of SAR to CAT can be understood with the help of (2. 7) and the projection-slice theorem. If an impulse could be transmitted, then (2. 7) indicates that the projections would be known exactly. For any other signal, the projections are effectively filtered. For example, a chirp signal [68] of a suitably large time-bandwidth product has a Fourier transform that has a magnitude which is nearly a rectangular function centered on the carrier frequency. If such a chirp were transmitted, then the known portion of the spectrum of the projection would be across the region of support of the chirp. If the radar could vary its look angle around 360°, the two-dimensional support of the Fourier transform of the ground patch would be a ring. However, in many instances, the radar can vary the look angle only a few degrees, so the Fourier information is known only over a small segment of the ring. Consequently, SAR can be viewed as a bandpass version of CAT.

It is also interesting to note the analogy between the finite (compressed) duration of the radar pulse and the finite detector width of CAT. The former effects bandpass filtering while the latter effects low-pass filtering, each by a convolution. Taking a cue from attempts in CAT to reduce the effects of finite detector width by deconvolving, perhaps it would be possible to compress the

radar pulse even more if the optimum noise property of the matched-filter receiver could be compromised.

With the observation of the similarity of SAR and CAT comes the possibility of using or modifying reconstruction algorithms from CAT in SAR. Such was done in [32], where it was found that convolution backprojection using linear one-dimensional interpolation gave results of about the same quality as direct Fourier inversion using interpolators with orders ranging from 8 to 18.

If the polar formatting that is indicated by the projection-slice theorem is honored in the reconstruction process, then the problem of target migration through resolution cells that was alluded to in the section on Doppler interpretation is automatically cured [65]. The convolution backprojection algorithm is one such algorithm.

The computational complexity of the convolution backprojection algorithm exceeds that of direct Fourier inversion even when the latter uses fairly high-order interpolators. However, convolution backprojection contains inherent parallelism which can be taken advantage of by systolic arrays [32]. (Favorable architectures exist for Fourier inversion, too.) In real-time applications, the convolution backprojection method can process each datum as soon as it is available. Thus, either the batch-mode latency of the direct Fourier inversion method is avoided or the peak speed requirement of the processor is reduced. There is the additional advantage that lower-quality images can be viewed before all of the data are available. In principle, this is also true for the direct Fourier inversion, although with a possibly substantial increase in computational burden. The work reported later in this dissertation may provide further inducement to use convolution backprojection-style algorithms.

2.2.5 Matched filter interpretation

The final interpretation of SAR to be discussed is that of the matched filter. This interpretation is not of the same status as the first three because it does not establish a model for the data collection process, only image reconstruction. It is not as highly developed a theory as the first three but seems to be a valuable conceptual tool and may inspire other reconstruction algorithms. It will be discussed here in relation to imaging a real-valued image at baseband, as in CAT. Some modifications would have to be made to adapt the model to coherent imaging at RF with a radar. A brief discussion is given in [60] and Chapter 6.

To obtain the desired result, a convenient property of the Radon transform will first be derived [22]. Define the Radon transform operator $\Re$ as

$$\Re\{f(x,y)\} = f_l(p,\theta).$$

Then the Radon transform of a function that is shifted, $f(\mathbf{x} - \mathbf{x}_0)$, where $\mathbf{x}_0 = (x_0, y_0)$, is given by

$$\Re\{f(\mathbf{x} - \mathbf{x}_0)\} = \int f(\mathbf{x} - \mathbf{x}_0)\, \delta(p - \xi \bullet \mathbf{x})\, d\mathbf{x}$$

$$= \int f(\mathbf{y})\, \delta(p - \xi \bullet \mathbf{x}_0 - \xi \bullet \mathbf{y})\, d\mathbf{y}$$

so that

$$\Re\{f(\mathbf{x} - \mathbf{x}_0)\} = f_l(p - \xi \bullet \mathbf{x}_0, \xi).$$

To obtain the matched filter result, the impulse response of the Radon transform is needed, since the reconstruction process is that of estimating the image at various points in the plane. By inspection, it is seen that

$$\Re\{\delta(x,y)\} = \delta(p), \tag{2. 8}$$

an impulse along the θ-axis in the Radon plane. [6] If the point $\mathbf{x}_0$ is expressed in the polar coordinates (r_0, θ_0), then

$$\xi \bullet \mathbf{x}_0 = x_0 \cos\theta + y_0 \sin\theta = r_0 \cos(\theta - \theta_0)$$

and

$$\Re\{\delta(\mathbf{x} - \mathbf{x}_0)\} = \delta(p - x_0 \cos\theta + y_0 \sin\theta, \theta)$$

$$= \delta[p - r_0 \cos(\theta - \theta_0), \theta].$$

This impulse response is a sinusoidal impulsive curve, the amplitude of which is determined by the distance of the point from the origin and the phase of which is determined by the angle of the point. The matched filtering operation for this signal, which will normally be combined with other such signals in $f_l(p, \theta)$, is to integrate along the same sinusoidal path. This operation, for a generic point (r,ϕ), is therefore

$$\int_{-\infty}^{\infty} \int_{0}^{2\pi} f_l(p, \theta)\, \delta[p - r\cos(\theta - \phi), \theta]\, d\theta\, dp\,.$$

Calculating the integral over p leaves the operation

$$\int_{0}^{2\pi} f_l[r\cos(\theta - \phi), \theta]\, d\theta\,. \tag{2. 9}$$

[6] An interesting observation made in passing is that (2. 8) represents a single backprojection, since the right-hand side is to be thought of as a function of two variables.

Identifying the quantity $r\cos(\theta-\phi)$ as the projected distance p of the point being reconstructed results in

$$\int_0^{2\pi} f_l(p,\theta)\,d\theta\,. \tag{2. 10}$$

This operation is seen to be essentially the same as backprojection, as in (2. 6). Note that (2. 6) is shown operating on a set of filtered projections. If the matched filter idea is used in the form shown in (2. 10), then other compensation must be made to obtain a focused image. This implies that the so-called rho-filtering or filter-of-backprojections [22] is required. On the other hand, (2. 10) can be seen as the integration of the point-by-point reconstruction described, for example, in [32]. This method will be discussed more in Chapters 4 and 5, for modified algorithms. The relevant point here is that the focused image can be reconstructed by computing the above integral along the indicated paths through the Radon plane *after* the filtering operation of (2. 5). This results in

$$f(x,y) = \int_0^{2\pi} \bar{f}_l(p,\theta)\,d\theta$$

which is identical to (2. 6) except for a scale factor which can be accounted for as normalizing the averaging operation implied by the integral, and by the upper limit of 2π instead of π, which is not conceptually important because it indicates the use of redundant information (see (2. 2)).

In summary, the matched filter model of image reconstruction leads to the same reconstruction formula as the convolution backprojection method. Also, there has been some work done in generalizing the Hough technique using spatial matched filtering ideas [53]. It remains to be shown whether the statistical assumptions that accompany matched filtering [69] apply fully to the noiseless case of inverting the Radon transform where the "noise" is the interfering projections from other points in the image as they sinusoidally weave their way around and across the projection of the point being reconstructed. (Comments similar to this are made in [18].) Until that time, claims about the optimality of such a reconstruction process would appear to be tenuous. However, it is useful as a conceptual tool if nothing else in forming a matched filter for detecting Doppler signals which are influenced by wavefront curvature and migration through resolution cells—this is one way of viewing the new algorithms of Chapters 4 and 5, even though this path is not developed in this dissertation.

As a final note, Walker [65] also comments on the matched filter idea.[7] He shows that with the

[7] Other authors also use the matched filter or two-dimensional correlation model as well, but it remains to rectify the work of these people with the tomographically-based interpretation given here and in [60]. The incentive for this kind of algorithm is tremendous when one considers the possiblitlies of optical digital signal processing [70].

usual plane wave assumption and polar formatting, a point target results in a recorded signal which is a linear diffraction grating, the spatial frequency and orientation of which depend on the polar coordinates in the x-y plane. He notes the possibility of using a matched filter to reconstruct the image, but states that a different filter is necessary for each point to be reconstructed. He comments that a more suitable method is provided by the combination of polar formatting and Fourier transformation. What he fails to mention is that Fourier transformation *is* the matched filter for sinusoids of unknown frequency and amplitude, a point which becomes clear upon examination of the Fourier integral and the casting of signals as members of a vector space. The Fourier transform computes an inner product of the signal being analyzed with all members of an ensemble of sinusoids.

2.3 Related Imaging Techniques

This section will briefly mention several imaging techniques which are related, some loosely, to SAR imaging. Little commentary is offered and no attempt is made to consider adapting any of them to the wavefront curvature problem or other "second-order" SAR problems, even though it may be fruitful to do so. Also, little attempt is made to unify or to interpret any of the related techniques.

The paper by Walker [65] and the related paper by Brown [66] are mainstream radar papers, albeit important ones. Their inclusion in this section is warranted by a detailed discussion in [65] of some of the aberrations that result when the spherical wavefront of the radiated field is taken into account and of a brief mention in [66] of a possible alternative to imaging under these conditions. The latter suggestion is to process the image in smaller sections so that the condition $R \gg L$ is met for each, where L is the maximum dimension of each subscene.

Optical and microwave holographic [71] techniques bear some similarity to SAR. In fact, it has often been observed that a recording of stripmap SAR data is a hologram in the cross-range dimension. The relationship between continuous-wave "holographic" radars and SAR could be clarified using Walker's range-Doppler imaging model [65].

Farhat and colleagues have worked in microwave imaging of conducting objects. Image formation is generally thought of as Fourier inversion based on the projection-slice theorem. Reference [72] reports on the determination of the shapes of three-dimensional perfectly conducting objects using frequency and angular diversity. Reference [73] describes novel hybrid optical-digital hardware for tomographic reconstruction using both coherent and noncoherent light. Reference [24] describes high-resolution microwave imaging using angular, spectral, and polarization diversities combined with knowledge of target symmetry resulting in speckle-free, centimeter-resolution imaging in the 6-17 GHz range.

Steinberg [74] describes theoretical and experimental results of imaging aircraft from the ground with resolution that is sufficient for target recognition applications. The images were made

using a self-calibrating, adaptive-beamforning scheme for distorted arrays and the radio camera equipment at the University of Pennsylvania [75]. Inverse SAR principles were included in some of the imaging methods.

Halevy [76] discusses two image reconstruction methods. The first is simply the convolution backprojection discussed here, based on the polar form of the two-dimensional Fourier transform written in polar coordinates, similar to (2. 4), for example. The second method is obtained by rearranging the order of the integrals so that integration over the angular variable in the Fourier plane is done first; this integration is actually a circular convolution which can be implemented by using a fast Fourier transform three times. The algorithm maps points on a circle in the Fourier space to points on a circle in the reconstructed image, i.e ., from the polar raster in the Fourier space to a polar raster in the image. The algorithm is claimed to be highly efficient due in part to the elimination of interpolation.

A striking new kind of range-Doppler imaging was proposed by Bernfeld [77] and Feig and Grünbaum [78], and improved upon by Snyder et al . [79]. This method depends on the ambiguity function of the transmitted signal (which is processed in a matched-filter receiver) being highly concentrated along a line in the delay-Doppler plane. An ambiguity function which has a two-dimensional Gaussian shape the level curves of which are highly eccentric ellipses is one such function. For a single pulse transmission, the received signal is the ambiguity function (which is the point spread function for that pulse) convolved in the delay variable with the target reflectivity. Therefore, to the extent that the ambiguity function approximates an impulsive ridge at some orientation $\theta + \pi/2$ radians, the returned signal has the form of (2. 3) with angular parameter θ, taking the form of a projection of the ground patch expressed in delay-Doppler coordinates. It is a property of ambiguity functions [18] that multiplication of the generating time-domain signal by a quadratic phase function shears the ambiguity function parallel to the Doppler axis, and that multiplication of the Fourier transform of the same signal by a quadratic phase function causes a shearing parallel to the delay axis; [77] seems to imply usage of the former and [78] uses the latter. These shearings resemble a rotation of the ambiguity function ([79] reports modifying the complex envelope of the generating function so that the ambiguity function is *rotated*), so that the projective information implemented by (2. 3) is found over a variety of angles. Reference [79] suggests a modification to correct for the ambiguity function not being uniform along its ridge by noticing a similarity to positron emission tomography. Perhaps the most remarkable aspect of these techniques is that no motion is required between the radar and the scene, the imaging depending only on modulation of the FM rate from pulse to pulse. It would appear that a more general imaging method could be had by applying other forms of pulse-to-pulse modulation, allowing for ambiguity functions that do not approximate an impulsive ridge, and applying corrections similar to those used in X-ray tomography and other areas to compensate for finite detector width.

Other related imaging techniques include electron microscopy and various acoustic imaging schemes [80]. An excellent collection of papers on imaging which contains several of those referenced here is [81].

CHAPTER 3
METHODOLOGY AND BASELINE SIMULATIONS

THIS CHAPTER describes the way in which the simulations for developing and testing new SAR reconstruction algorithms were conducted. The first application of this methodology is the calculation of the Radon transform of a specially-selected test function and its inversion using a standard method. One measure of the new algorithms, described in later chapters, is how closely they are able to approximate the baseline reconstruction of this chapter.

3.1 Non-Bandlimited Imaging

Perhaps the most significant item of methodology to be discussed is the relaxing of the known bandlimited aspect of SAR in order to aid the simulations. First, the acceptability of this will be discussed, and then the necessity.

As shown in Section 2.2.4, the bandlimiting nature of SAR is due to two factors. Bandlimiting in the radial direction in the Fourier plane of the ground patch is due to the finite bandwidth of the transmitted pulse—there is energy only from a finite range of frequencies impinging on the ground patch, and information only from that band can be known. Bandlimiting in the angular direction is common; this is due to the radar varying its look angle over only a part of a full circle. In this dissertation, it is assumed that an impulse is transmitted and that the radar (or the midpoint of a bistatic radar) circumnavigates the ground patch.

Justification for not simulating the bandlimiting is provided by [82] which was motivated by the observation that high-quality images are obtained by imaging systems (SAR, holographic) which are able to collect data only from a very restricted portion of the Fourier plane. The scene reflectivity g can be factored into magnitude and phase parts according to

$$g(x, y) = m(x, y) \exp\left[j\phi(x, y)\right].$$

If the phase is highly random, then the Fourier transform of $\exp(j\phi)$ will be very broad, so that the Fourier transform of g, which is obtained by the convolution of the transforms of m and ϕ, will contain magnitude information which is distributed over a large portion of the Fourier plane. The phase is seen to modulate the magnitude information, spreading it over a larger region than it would otherwise cover. This process is similar to spread spectrum modulation in some communication systems. Reference [82] shows that when real-valued images are reconstructed from frequency-offset Fourier data, edges at various orientations are lost, depending on the location of the

data in the Fourier plane. However, when the same images are converted to complex-valued functions by the addition of a highly random phase function, computer simulations show reconstructions of good quality. A statistical analysis shows that under these conditions, image quality is not affected by the location of the Fourier data that are used but only by the size of the band that is used. The simulations shown there support the statement that the only significant effect of reconstructing images from frequency-offset data is some loss of resolution which is consistent with the size of the band used, but that edge artifacts are absent.

Some advantages accrue from analyses and simulations which assume a transmitted impulse. The results so obtained are not restricted to the peculiarities of a specific waveform. (For example, if a suitable chirp signal is transmitted, the received signal is nearly the Fourier transform of the projections of the ground patch [26], not the projections directly.) If the radar data collection process is thought of as a series of cascaded subsystems starting with an impulse generator operating at the PRF and including, for example, the transmitter, the transmitting antenna, outgoing propagation effects, the ground patch, incoming propagation effects, the receiving antenna, and the receiver, then in order to specialize the results obtained using an impulse it is only necessary to add at the correct point in the chain a box the impulse response of which is the desired pulse. Historically, the chirp is by far the most popular radar signal when high range resolution is desired, since it is relatively easy to generate and to matched filter. However, with advances in transmitter and digital technology, other signals with more favorable characteristics such as ambiguity functions are becoming more feasible. Indeed, the concept of adaptive radar is being discussed [83].

There is another advantage connected to algorithms which are designed with full-circle look angle variation in mind. If an algorithm can make a high-quality reconstruction with all of the distortions which are present in non-plane-wave data which are collected from all angles, then surely this is a conservatively-designed algorithm when only data from a restricted angle are available. One could argue that using a high-quality algorithm is a waste of resources when a lesser one will suffice, but perhaps an engineering approach should be to start with the better algorithm and simplify it if some performance can be sacrificed.

The necessity of the preceding methodology will now be discussed. When working with imaging systems, the most important property is the impulse response. Due to the discrete nature of the sampling within a single projection and the discrete nature of the sampling in look angle, an impulse will almost always fall between samples unless special care is taken in placing it, i.e ., it will “fall through the cracks.” Apparently, the only place where this will not happen without significantly modifying the sampling strategy is at the origin of the scene. Even then, special care would have to be taken to ensure that a sample (line integral) is taken exactly through the origin for every projection. With slightly more complexity in the sampling method, an impulse placed anywhere in the plane would be adequately sampled, but this might throw the validity of any subsequent reconstructions into question, since it would be reasonable to wonder how well impulses

that did not receive such special treatment would be reconstructed. This technique would almost certainly be limited to testing one impulse at a time.

In SAR simulations that assume plane waves, the impulse response is easily obtained, if indirectly. This is possible by computing the Fourier transform of an arbitrarily-placed impulse at the desired points. By the Projection-Slice theorem, this yields the same information as if the projections of the impulse could be calculated directly, and the problem is neatly circumvented.

In SAR simulations that do not assume plane waves, the absence of the Projection-Slice theorem dictates that other means must be used. With the problems presented above, it seems that examining the impulse response of non-plane-wave algorithms is not tractable. (In addition, some reconstructions shown later have spatially-varying impulse responses.) One alternative approach would be to use some other (complex-valued) test function that had a highly random phase function so that the modulation process would aid the reconstruction. (One may wonder if it is valid to extend the random-phase argument to the non-plane-wave case. It would appear so, since actual SARs do not place exact plane waves over the ground patch, yet achieve high-quality reconstructions—it is engineers, not Nature, that have the penchant for plane waves.) This approach was not taken here because of the numerical problems and approximations associated with computing projections through a test function that is defined on pixels. It was feared that there would be more artifacts of the projection computation in the reconstruction than artifacts of the algorithms. At least, it would be hard to separate the two. Fortunately, the argument of random phase modulation allows the use of a real-valued test function, the projections of which can be computed in closed form. This follows from the fact that the only significant difference between reconstructing a real-valued function from a full-circle set of full-band projections and reconstructing a random-phase function from a restricted set of bandlimited projections is that the latter will show reduced resolution.

The test function selected consists of four right circular truncated cylinders, shown in Fig. 3. 1. The parameters of these cylinders, or "top hats," have been selected in order to provide a worthwhile measure of the quality of algorithms. The narrow top hat is only two pixels in radius, about the smallest diameter allowable for the sampling intervals used (discussed below), and roughly useful as an approximation to an impulse. It is placed very near to the edge of the ground patch, here a circle which fits just inside the square base of the plot. The large top hat is used to simulate an extended target, especially useful when an algorithm is shift-variant, or suspected to be so. The indentation in the top of this top hat is a negative-going top hat, useful when observing any overshoots which would appear as undershoots on a positive-going top hat and possibly be obscured by the hidden line removal of the plotter. The remaining top hat is of intermediate radius and modest amplitude, placed near the large top hat to detect interference from it. The test function is thought of as being of radius $L = 63$ pixels so that there is a pixel at the origin and the diameter is 127 pixels—by padding with only one zero column and row, the image could easily be

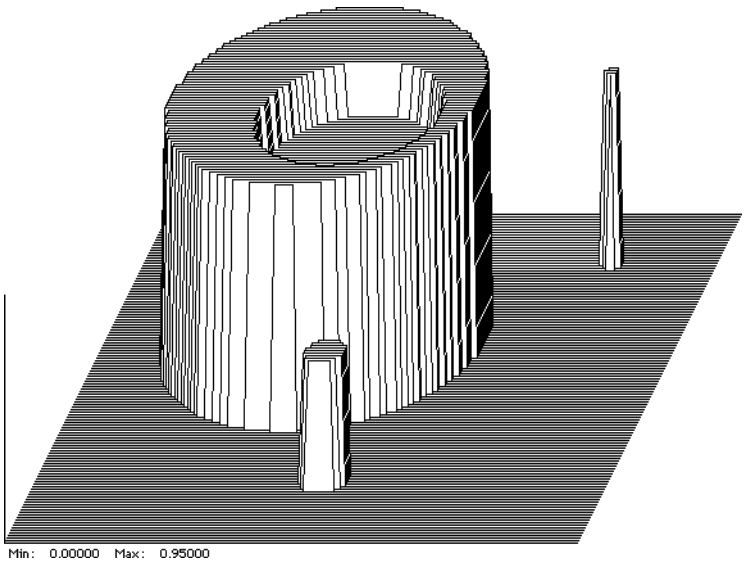


Fig. 3. 1. The function that is used to test various reconstruction algorithms.

processed with a 128 by 128 fast Fourier transform (FFT), if desired. The parameters of the test function top hats are given in Table 3.1. Distance units are pixels and angular units are degrees from the x axis.

The choice of top hats to make up a test function allows a variety of projections to be calculated in closed form, avoiding numerical integration or the inaccuracies associated with a test function that is defined pixel-by-pixel. A possible problem with this test function is that Radon's inversion formula depends on the continuity of g and the continuity of the partial derivative of the projections $f_l(p,\theta)$ with respect to p [84]. Both of these assumptions are violated here. However, test functions like this are often used (see, for example, [34]) with little or no ill effects, and indeed the same will be observed in later reconstructions shown here.

3.2 Simulation Details

This section merely lists the parameters of the simulations which remain constant throughout. Both the ground patch and the reconstructed image have a radius of 63 pixels and a diameter of 127 pixels, as mentioned earlier. All reconstructions are made from 198 projections. This number was deemed adequate and is comparable to numbers used in other work reported in the CAT literature. There obviously are questions concerning sampling issues, and [46] presents some novel work in sampling the Radon transform. Unfortunately, this work is not directly applicable to the present problem. However, during most of the time that the new algorithms were being developed, early reconstructions were made from 100 projections simply to speed the process along. The images so reconstructed appeared somewhat more grainy than those which were later made using 198 projections, but no other artifacts that could be attributed to undersampling were evident. Each

TABLE 3.1
PARAMETERS OF THE TEST FUNCTION

Radius	Height	Distance from Origin	Angle from x-axis
35	0.95	20	135.9
2	0.75	60	47.0
20	-0.20	10	135.9
5	0.50	40	89.1

projection consists of 127 samples. It is not necessary to have the number of samples per projection the same as the width of the ground patch and reconstructed image in pixels, but it sometimes simplifies the writing of computer programs. (A case where there seems to be no simplification is in bistatic SAR simulations.)

As described below, the filtering of the projections is done with an FFT, and at that point a Hamming window is applied to smooth the reconstructions, at the expense of some resolution. (Perhaps some artifacts caused by violating the continuity conditions are smoothed over as well.) On backprojection, linear interpolation is used to get the backprojected value at pixels between known samples of the filtered projections. Quadratic interpolation was tried with some improvement in image quality, but by and large, numerical issues (and there are many) were not studied—only the simplest or most obvious numerical methods were used so that more effort could be spent on the central issues of algorithm development.

Some comments about how the reconstructions are plotted are in order. There is no attempt made to correct for shifts in the mean value of the reconstructed images which can occur since the filtering operation has zero transmission at zero frequency [37]. There is evidence of this in only a few cases, and then it is slight and easily compensated visually. The vertical scale on the three-dimensional plots is linear in order to more accurately show what the eye would see if a gray scale image were shown. Also, a decibel scale would exaggerate low-level phenomena which are frequently dominated by artifacts that result from the use of simple numerical methods, as discussed above. The plotter removes hidden lines and automatically scales the plots according to the minimum and maximum data points, fitting the data into a fixed display range. The only exception is if there are no negative data, which can occur in a high-quality reconstruction with a slight level shift. In this case, the minimum is taken to be zero. In all cases, the region outside the circle representing the original scene is set to zero, and the data minimum and maximum are printed at the bottom of the image. The x axis points from left to right and the y axis points from front to back. The data points are connected by a straight line (except for screen "jaggies"), so vertical walls such as seen in Fig. 3. 1 will appear with finite slope, with less such distortion on less steep walls. Contour plots of all of the reconstructions are also provided. These are helpful in visualizing certain details which are not obvious on the three-dimensional plots. Most contour plots are made with 20

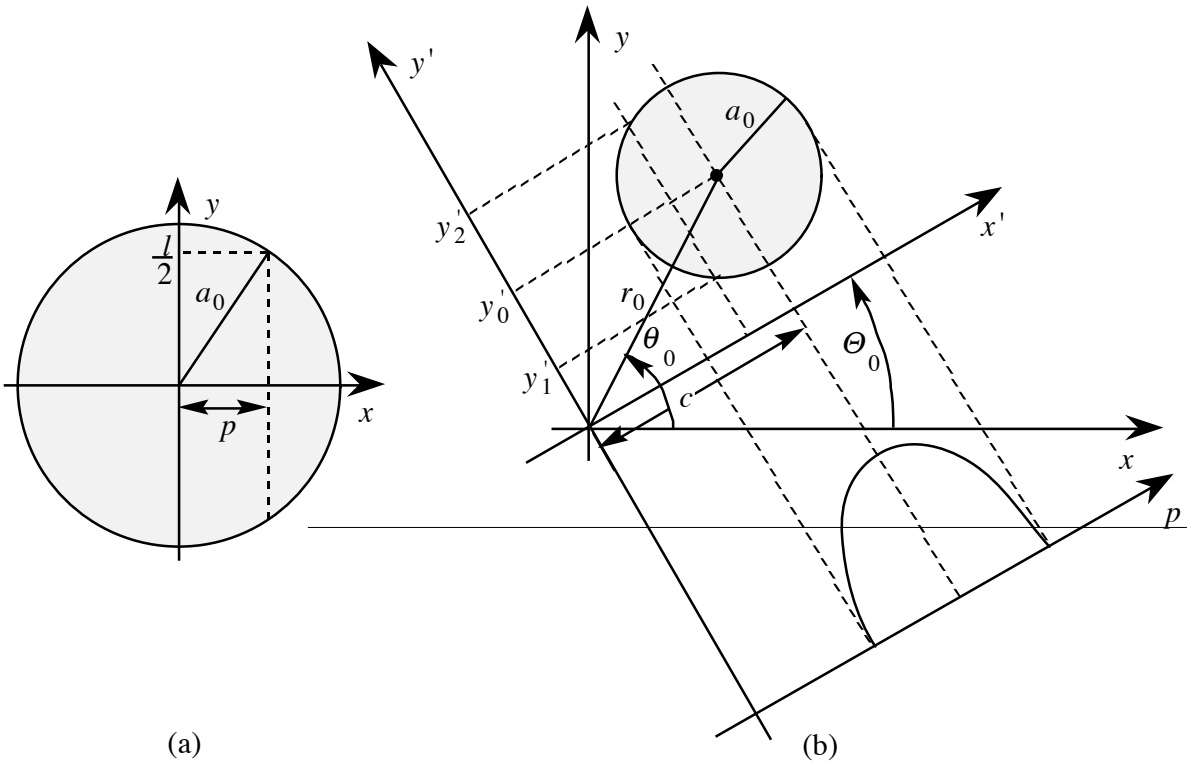


Fig. 3. 2. Geometry to aid in computing straight-line projections of a top hat function. (a) Top hat centered at the origin. (b) General top hat.

contours uniformly spaced between the minimum and maximum of the image data points. Three-dimensional perspective plots and contour plots are also given for the projections. Here, only half of the projections are shown in perspective, to improve clarity.

3.3 Baseline Simulations

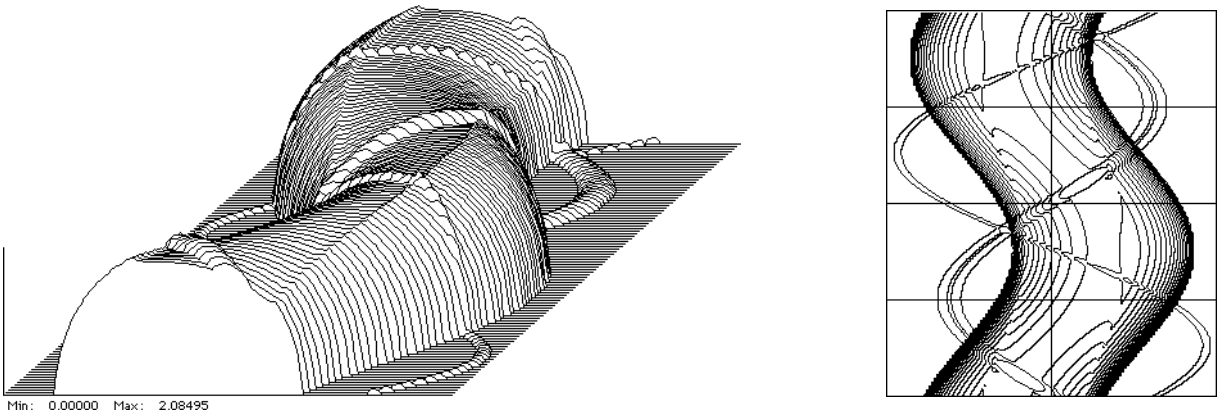
The results of this section are provided as a comparison for later results using new algorithms. The style of presentation, which is carried into the following chapters, is to show the calculation of the projections for the test function of Fig. 3. 1, then to describe the algorithm, then to show some reconstructions. In Chapter 5, the last two activities are combined as iterative attempts are made to improve the algorithm.

3.3.1 Calculation of straight-line projections

The projection of a top hat function will be found in two steps. In Fig. 3. 2 (a), the length l of the vertical line segment a distance p from the origin is seen to be

$$l = 2\sqrt{a_0^2 - p^2}$$

where a_0 is the radius of the top hat. For the general top hat $f(x, y)$,



(a) (b)

Fig. 3. 3. Straight-line projections (Radon transform) of the test function. (a) Perspective plot. The horizontal axis is p and the depth axis is Θ. (b) Contour plot. The horizontal axis is p and the vertical axis is Θ.

$$f(x,y) = \begin{cases} 1 \text{ for } \sqrt{(x - r_0\cos\theta_0)^2 + (y - r_0\sin\theta_0)^2} \le a_0 \\ 0 \text{ otherwise} \end{cases}.$$

As shown in Fig. 3. 2 (b), it is necessary only to find c, the offset of the top hat projected onto the p axis. This quantity is

$$c = r_0 \cos(\theta_0 - \Theta)$$

so the straight-line projections (Radon transform) of a top hat are

$$f_l(p,\Theta) = \begin{cases} 2\sqrt{a_0^2 - (p-c)^2}\ , & |p-c| \le a_0 \\ 0\ , \text{ otherwise} \end{cases}$$

or

$$f_l(p,\Theta) = \begin{cases} 2\sqrt{a_0^2 - [\,p - r_0\cos(\theta_0 - \Theta)]^2}\ , & |p-c| \le a_0 \\ 0\ , \text{ otherwise} \end{cases}. \qquad (3.\ 1)$$

This transform is shown in Fig. 3. 3 for the test function. Such plots are sometimes called sinograms, especially when plotted on a gray scale, since small features in the scene transform into sinusoidal traces.

In practice, only samples of (3. 1) are known and processed, and one may wonder about the suitability of the test function in terms of the bandwidth of its projections. The two-dimensional

Fourier transform of a top hat function $f(x, y)$ of radius a_0, unit height, and centered at the origin, is

$$F(\Omega_1, \Omega_2) = 2\pi a_0 \frac{J_1\left(a_0\sqrt{\Omega_1^1 + \Omega_2^2}\right)}{\sqrt{\Omega_1^1 + \Omega_2^2}}$$

where J_1 is the Bessel function of order one. This is circularly symmetric, and the Projection-Slice theorem gives the Fourier transform of a projection of $f(x, y)$ as

$$\mathsf{F}\{f_l(p, \Theta)\} = 2\pi a_0 \frac{J_1(a_0\rho)}{\rho}$$

in the transform variable ρ. The Fourier transform of an offset top hat has the same magnitude, but a different phase function that is commensurate with the projected offset c. The smallest top hat in Fig. 3. 1 was used specifically to stress any algorithm that is sensitive to non-bandlimited data. While no rigorous justification can be given, no serious artifacts that could readily be attributed to aliasing were noticed in any of the reconstructions, although some low-level effects are certainly present for which full accounting is not had. Again, such test functions are common in the literature, and a study of aliasing effects and oversampling is obviously an area for more work. In practice, the projection data are low-pass filtered before sampling by the finite detector size in CAT and presumably by finite aperture times in analog-to-digital converters in SAR. In any event, the use of a smaller-bandwidth test function would only make the reconstructions in this dissertation appear better.

3.3.2 Convolution-backprojection image reconstruction

The details of the convolution-backprojection algorithm used for the baseline reconstructions can be found in [36]. Briefly, a modified convolution kernel, relative to (2. 5), is used. This is defined in terms of its Fourier transform $H(\rho)$ as

$$H(\rho) = |\rho| b_\pi(\rho)$$

where

$$b_\pi(\rho) = \begin{cases} 1, & |\rho| < \pi \\ 0, & \text{otherwise} \end{cases} .$$

Here, it has been assumed that the projectional sampling interval is one. The convolution kernel then is

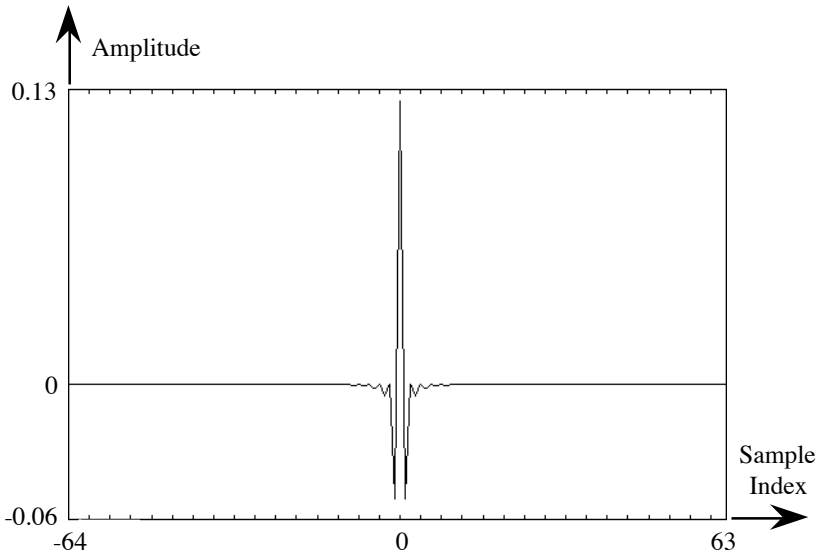


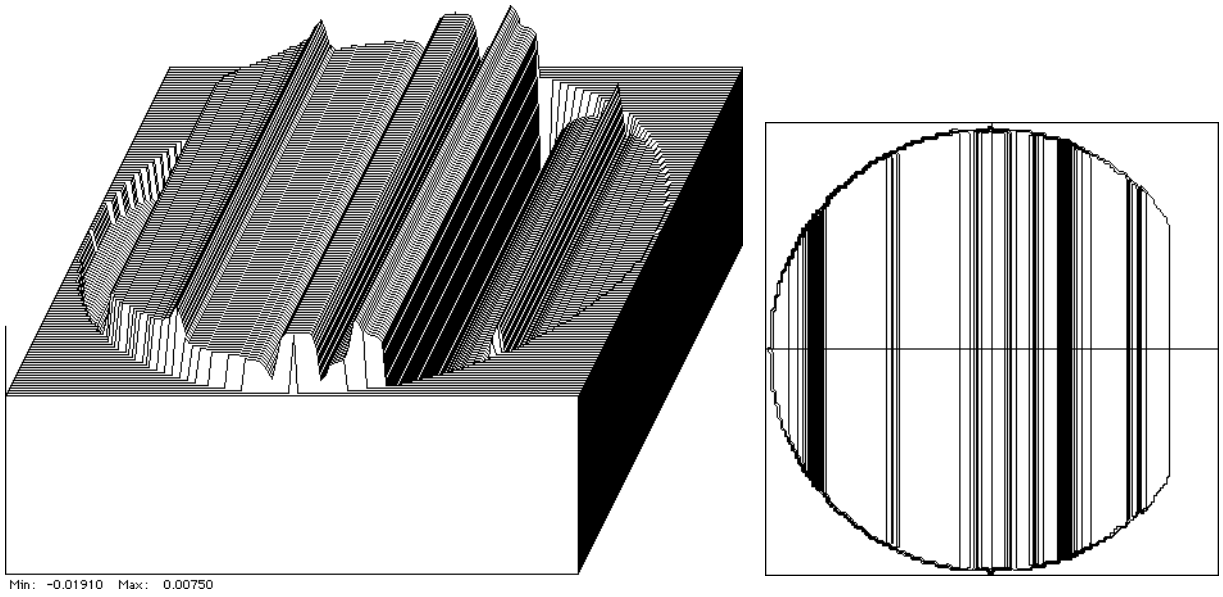
Fig. 3. 4. The sampled impulse response of the filter in convolution-backprojection image reconstruction.

$$h(p) = \frac{1}{2}\frac{\sin(\pi p)}{\pi p} - \frac{1}{4}\left[\frac{\sin(\pi p/2)}{\pi p/2}\right]^2, \tag{3. 2}$$

samples of which are

$$h(n) = \begin{cases} \frac{1}{4}, & n = 0 \\ 0, & n \text{ even} \\ -\frac{1}{n^2\pi^2}, & n \text{ odd} \end{cases} \tag{3. 3}$$

for integer values of n. This function is plotted in Fig. 3. 4 for 128 samples. The convolution operation is done in the frequency domain using an FFT. (In some literature, algorithms which use frequency-domain filtering are called filtered backprojection algorithms, but the distinction seems needless.) In order to be compatible with the usual indexing of FFTs, the impulse response (3. 3) is shifted to the right by 64 samples. It is then zero-padded to length 256, discrete Fourier transformed, and the result is multiplied by a Hamming window. The last operation has a large effect on the frequency response of the filter and consequently the actual impulse response is different than that of (3. 3). Each projection is also zero-padded to 256 samples, transformed, multiplied by the modified filter, and inverse transformed. The indexing is adjusted to be compatible with



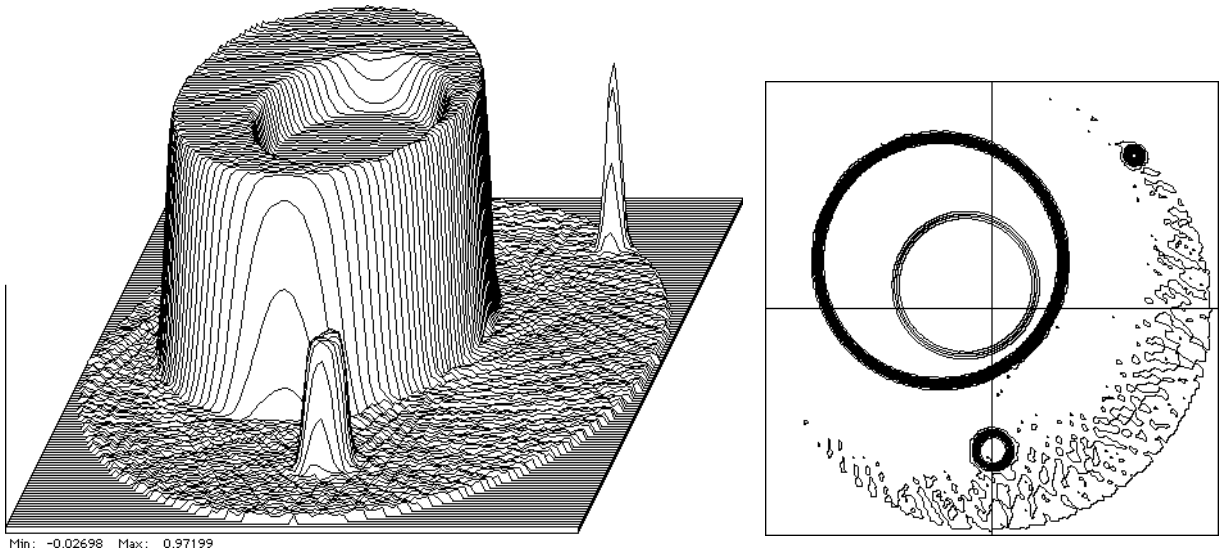
(a) (b)

Fig. 3. 5. Backprojection of a single filtered projection of the test function. Original projection angle was zero degrees. (a) Perspective plot. (b) Contour plot.

image pixel coordinates. Then each filtered projection is backprojected using linear interpolation to pixel locations, and accumulated to form the final reconstructed image. All the details can be found in the listings of Appendix D.

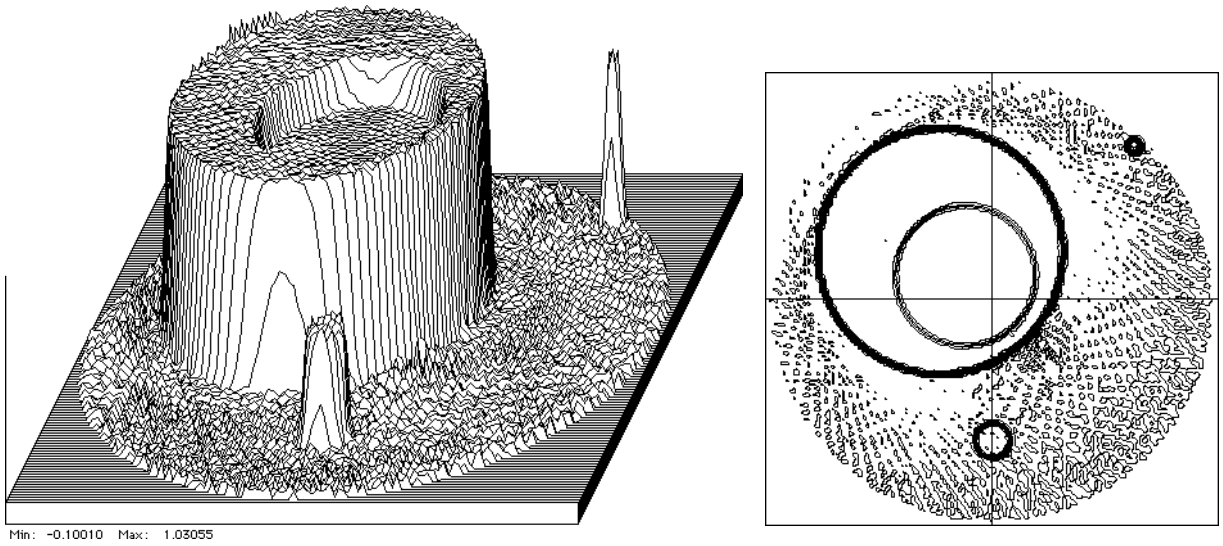
A plot of a single backprojected filtered projection of the test function is shown in Fig. 3. 5. The one shown is for the projection which results when the radar is at zero degrees, measured from the positive x axis. One can see prominent positive-valued features which backproject through the various top hat locations. These positive features and others from other backprojections will constructively interfere to reform the top hats in the final image, while the negative features act to "chisel away" the unwanted positive contributions from the other projections in precisely the right way. This view of the reconstruction process is very instructive.

The complete reconstruction is shown in Fig. 3. 6. This reconstruction will serve as the baseline for many other reconstructions in this dissertation. The extended portions of the two widest top hats are quite flat, while the smallest top hat is well-concentrated, but with some widening evident at the base. As will be seen below, the Hamming window is largely responsible for the reduced slope of the steep walls. All four top hats are reconstructed with very nearly the correct height. The floor appears somewhat noise-like, but if finer increments were used in the contour plot, a rich structure of low-level streak artifacts would be revealed. The corners of the plot, outside the circular scene, are set to zero, and this serves as a reference for the vertical scale, which is shown at the lower left corner of the perspective plot. Due to the automatic scaling of the data by the plotter, this plot shows that there is at least one small negative value somewhere in the image.



(a) (b)

Fig. 3. 6. Reconstruction of the test function from its straight-line projections using a conventional convolution-backprojection algorithm. This figure is to be used as a base-line for comparison for other reconsructions. (a) Perspective plot. (b) Contour plot.

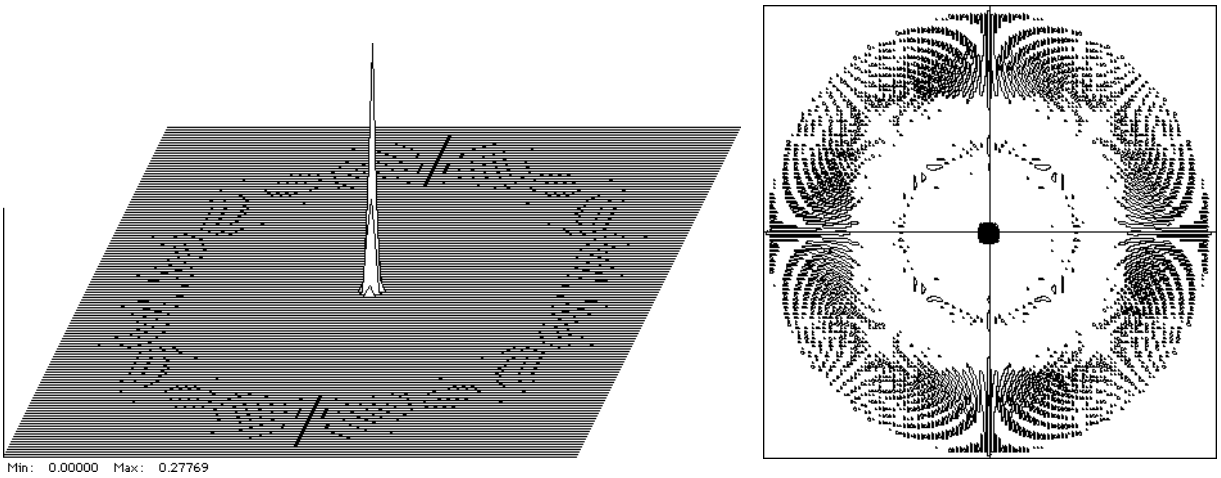


(a) (b)

Fig. 3. 7. Reconstruction from straight-line projections using conventional convolution-backprojection, only without a Hamming window. (a) Perspective plot. (b) Contour plot.

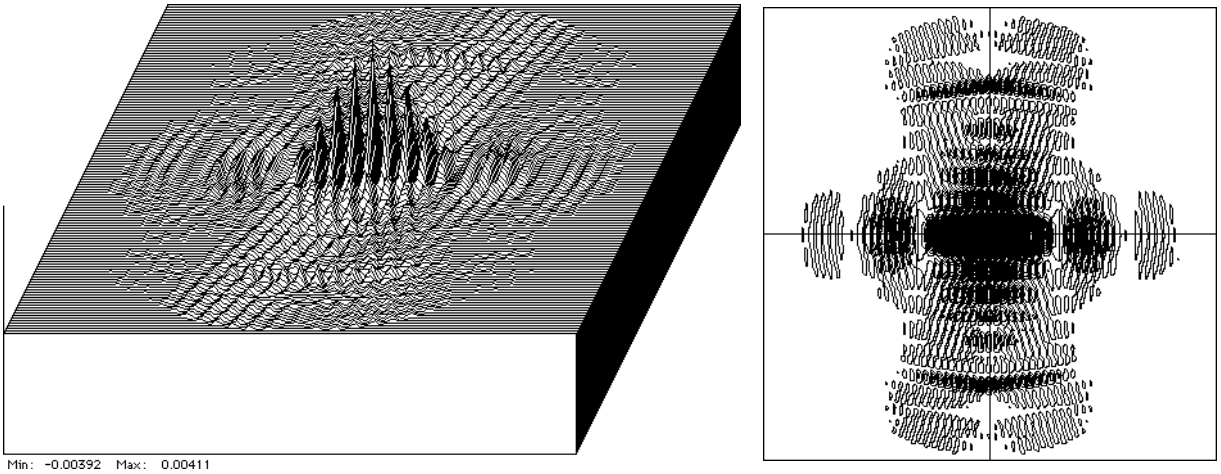
Experience with this and other reconstructions indicates that this undershoot most likely occurs at the base of the large top hat.

To study the effects of the Hamming window, Fig. 3. 7 is provided. As expected, the "noise" is greater, including some overshoots at the tops of major features. Also, the walls are steeper,



(a) (b)

Fig. 3. 8. Reconstruction of an impulse from its straight-line projections using a conventional convolution-backprojection algorithm. (a) Perspective plot. (b) Contour plot (500 levels).



(a) (b)

Fig. 3. 9. Reconstruction of a central impulse from bandlimited data. (a) Perspective plot. (b) Contour plot (100 levels).

showing that, in addition to the plotting artifact mentioned earlier, the Hamming window is largely responsible for this effect.

As another example of a reconstruction using this algorithm, Fig. 3. 8 shows the result of reconstructing an impulse located at the origin of the scene. (The Radon transform of this impulse is an impulse sheet, $\delta(p)$, uniform in Θ.) All computed projections include one line integral through the impulse. Fig. 3. 8 (b) deviates from usual practice in showing 500 contour levels, since the level of the area surrounding the peak is so low.

Finally, Fig. 3. 9 shows a central impulse reconstruction from bandlimited data, the only such bandlimited case which is practical for comparison with reconstructions in later chapters. Perhaps generous according to some SAR applications, the variation in both frequency coordinates in the Fourier plane is close to 10%. Specifically, the variation in look angle extends over 19 of the 198 projections, centered on the x-axis, i.e., around $\Theta = 0$. The radial frequency is limited by setting all but 26 of the bins of the DFT of the filter kernel to zero, effecting a rectangular window in the DFT domain. Those bins which were left unaffected are centered on $\pi/2$ and 3 $\pi/2$ on the usual normalized cyclic frequency scale, namely 57…69 and 185…197 where the filtering is over frequency samples numbered 0…255. A comparison with reconstructions, not shown, for which the frequency domain support was first limited to the wedge described above, and then to the annular ring described above, shows that by and large the straight-line artifacts which are arrayed a few degrees on either side of the y-axis are due to the restriction in look angle and the circular artifacts are mostly due to the restriction in radial frequency. Ordinarily, SAR image reconstruction would slide the piece of Fourier data so that it is more or less centered on the origin before inverting, affecting a modulating downconversion, but this was not done here. This accounts for most of the vigorous ocillations in the reconstruction.

CHAPTER 4
MONOSTATIC SAR

AS EXPLAINED in Chapter 1, in order to maintain good resolution across the reconstructed image when using plane-wave-based algorithms, the condition $R \gg L$ must exist during data collection, where R is the distance to the radar from the center of the ground patch and L is the radius of the ground patch. This claim is further supported in analyses presented in [65] and [26] and by reconstructions shown in this chapter.

A detailed analysis of the effects of wavefront curvature is given in [65]. There, the plane-wave analysis is given first, using a common approximation. The distance from the radar to a point on the ground is written exactly and then expanded into a binomial series, from which the first two terms are retained. Later, in the wavefront curvature analysis, an additional term from the expansion is kept, and the result for the image resolution for a polar format algorithm (such as convolution backprojection or direct Fourier inversion from well-interpolated frequency-domain data) is given as

$$\rho_{\text{polar}} \geq \sqrt{\frac{\lambda L^2}{16R}} ,$$

in which λ is the wavelength corresponding to the center frequency of the transmitted signal. For a fixed radar distance, the minimum resolution increases with the patch size. Cases are examined in which an astigmatic focus error is present, and the apparent (to a plane wave) positions of scatterers vary with look angle. These effects are seen to be dependent upon the position of the scatterer and the position of the nominal look angle (for restricted look angles), and are seen to be repairable by focusing the data processor. This task is stated as being "generally difficult to implement." It will be shown in this chapter that within the unconventional signal processing framework presented, the task can actually be fairly easy, although requiring somewhat more computation than Fourier-based methods which use an FFT but do not correct for wavefront curvature.

More effects of wavefront curvature are studied in [26]. From the requirement that all signals that are returned from each point be coherent, it is found that the maximum patch size is limited by the approximate relationship

$$L \leq \sqrt{\frac{4\lambda R}{8 \sin(2\theta_{\text{M}})}} \tag{4. 1}$$

TABLE 4.1
MAXIMUM GROUND PATCH SIZE L_{max}, FROM (4. 1)

λ	R	θ_M	L_{max}
0.03 m	30,000 m	3°	65 m
0.03 m	400,000 m	3°	240 m
0.03 m	30,000 m	1°	114 m
1.00 m	400,000 m	3°	1380 m
1.00 m	400,000 m	1°	2390 m

where $2\theta_M$ is the variation in look angle, assumed to be less than $\pi/4$, and, to simplify the derivation, it was assumed that $R \gg L$ still. This last restricts the general applicability of the result; however, Table 4.1 shows some maximum ground patch sizes L_{max} for typical parameters for airborne and spaceborne SARs, and the condition appears to be valid over a wide range of situations. In fact, Table 4.1 shows what may seem to be surprisingly small patches that can be properly imaged, according to this criterion.

While (4. 1) is based on coherence requirements, another requirement from [26], although usually less stringent, will be more useful for present purposes. The derivation here will deviate from that of [26] in that the condition $R \gg L$ is not used, and the result is extended slightly relative to the earlier paper. The quantity to be found is the error in range over the target field; reference to Fig. 4. 1 will be helpful. The error at the point (x, y) is

$$e_1(x_0, y_0) = \left[(R - x_0)^2 + y_0^2 \right]^{1/2} - (R - x_0) \tag{4. 2}$$

for the sufficiently general case of the radar being at $(R, 0)$. Inspection of Fig. 4. 1 shows that for fixed x_0 the maximum error occurs at the edge of the circular ground patch. Evaluating the above expression at the edge, $y_0^2 = L^2 - x_0^2$, gives

$$e_2(x_0) = \left(R^2 - 2x_0 R + L^2 \right)^{1/2} - R + x_0 \ .$$

The maximum of this function can be found to be at

$$x_M = \frac{L^2}{2R},$$

just to the right of the origin, as may be inferred from Fig. 4. 1. The value of the maximum is found to be

$$\max e_2(x_0) = \frac{L^2}{2R}. \tag{4. 3}$$

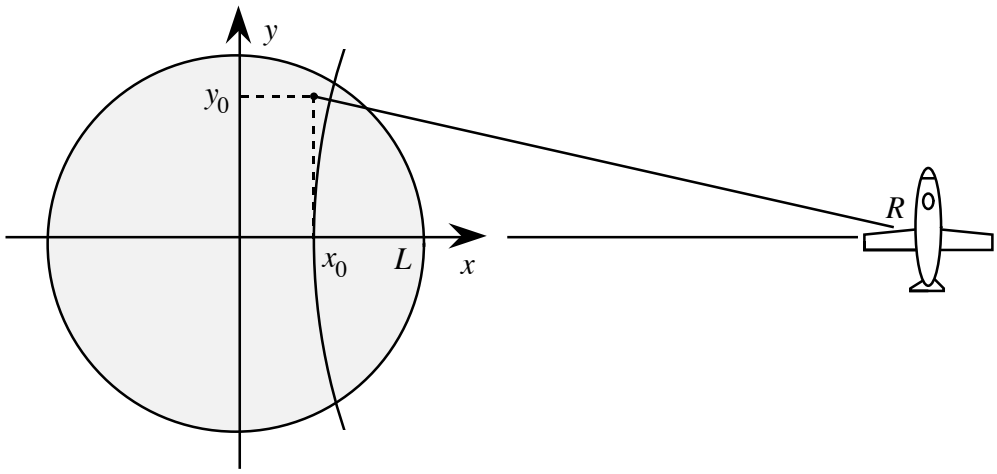


Fig. 4. 1. Geometry for deriving error in range over the ground patch under the plane wave assumption.

The error (4. 2) is plotted in Fig. 4. 2 over the ground patch for the parameters $L = 63$, $R = 72$, both in linear units of pixels, the value of R being typical of later reconstructions; the presentation is the same as for reconstructions, and it is hoped that this will not be found to be confusing.

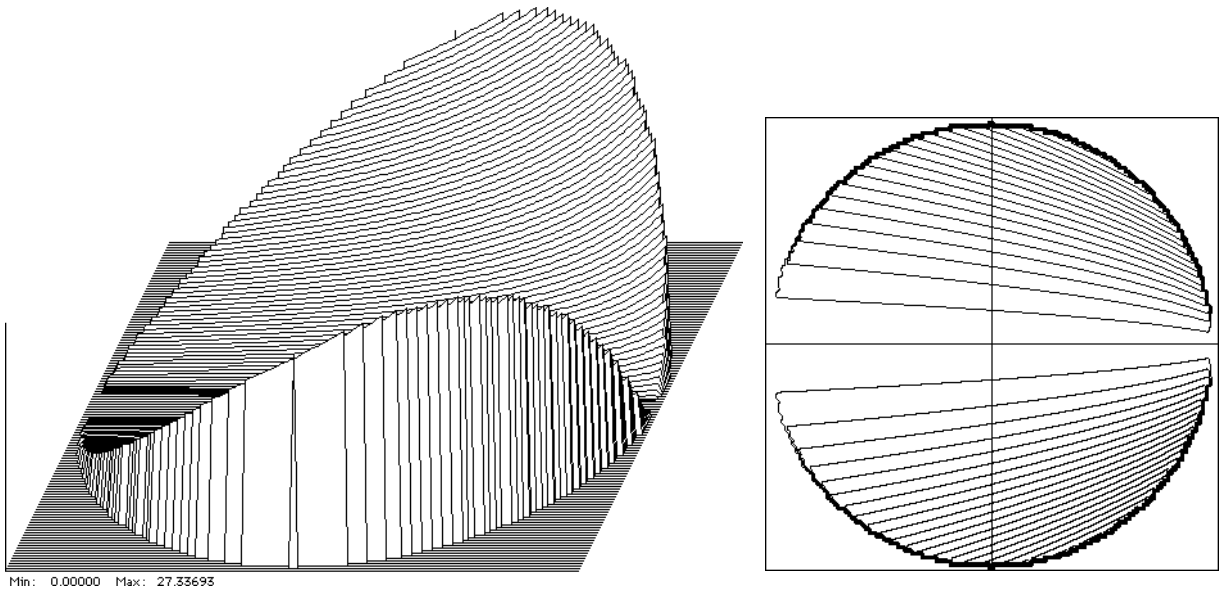
The above results can be used to estimate the amount and type of defocusing that will result if a plane-wave-based algorithm is used under significant conditions of wavefront curvature. Assume that convolution backprojection is used to reconstruct a point target at (x_0, y_0). Each backprojection is the filter impulse response, e.g ., (3. 2) or as such would be modified by a Hamming window. The central peak of each backprojection would miss the location of the point target according to

$$e_\Theta(x_0, y_0) = \left[\left(R - x_0 \cos \Theta - y_0 \sin \Theta \right)^2 + \left(- x_0 \sin \Theta + y_0 \cos \Theta \right)^2 \right]^{1/2} - \left(R - x_0 \cos \Theta - y_0 \sin \Theta \right)$$

which is merely the error (4. 2) with the target location expressed in terms of the rotating (x', y') coordinate system which has its x'-axis pointing through the radar, as in Fig. 2. 5. (The coordinates of Fig. 4. 2 may be regarded as x' and y', so that the figure rotates with the radar.) The radar angle which will cause the greatest error is

$$\Theta_M = \tan^{-1}\left(\frac{y_0}{x_0}\right) - \tan^{-1}\left(\frac{y_M}{x_M}\right)$$

and the angle which will cause the least error is



(a) (b)

Fig. 4. 2. Range error over the ground patch due to a plane wave assumption. The radar for this example is at coordinates (72, 0) in pixel units. The ground patch has a radius of 63 pixel units.

$$\Theta_{\mathrm{m}} = \tan^{-1}\left(\frac{y_0}{x_0}\right).$$

In fact, this angle corresponds to zero error, since the point being reconstructed lies precisely in the valley of the error function. A point located at the origin will be reconstructed exactly. Most radars operate so that the ground patch that is imaged lies near the origin of Fig. 4. 2, where the error is small. The approximate "size," then, of the reconstructed point is

$$\frac{x_0^2 + y_0^2}{R}$$

which is twice (4. 3) with L^2 replaced with $x_0^2 + y_0^2$. The factor of two is required since projections from Θ and $\Theta + \pi$ cause errors on opposite sides of (x_0, y_0).

The above observations are alternate ways of considering the focus problems and point migration discussed in [65]. Also, some light is shed on the need for focusing operations in the data processor which vary with pixel location and look angle. For limited look angle reconstructions, these effects may be manifested not only as focus errors but as registration errors as well.

In principle, a reconstruction algorithm could be devised as follows. The impulse response of a plane-wave algorithm could be found at each pixel in the image plane. Then, a filter with a spatially-varying impulse response could be made such that each pixel is properly focused. The algorithm would proceed in two steps, the first being the plane-wave algorithm, the second being the

focusing filter. The filter would in general have to operate over the entire image plane for each pixel, since information from each resolution cell would be scattered everywhere.

The algorithm presented in this chapter accomplishes this with much less fuss and requires little more computation than convolution backprojection. First, details concerning antenna fields will be discussed.

4.1 The Far Field

The purposes of this section are to review the concept of the far field of radiating sources so that the salient features may be commented on, and to clear any confusion (if necessary) that the reader may have to the effect that the far field might be a plane wave. The presentation is that of [85] and is two-dimensional, in keeping with the practice here of suppressing the height dimension. Also shown in [85] is the three-dimensional case. The main difference for the purpose at hand is that the attenuation in the latter case goes as $1/r$ instead of $1/\sqrt{r}$, in accordance with the conservation of energy. The notation in this section is not intended to be consistent with that of other sections, especially with respect to coordinate systems.

A plane wave uniform in z (the height direction for the radar problem) which propagates from an aperture at $x = 0$, at an angle α relative to the x axis, has a y-component

$$E_y(x, y) = A(\alpha) \exp\left[-jk(x \cos \alpha + y \sin \alpha)\right]$$

where k is the radian wavenumber and $A(\alpha)$ is the amplitude. By letting $k_y = k \sin \alpha$, $k_x = k \cos \alpha$, and $F\left(k_y\right) = \lambda A(\alpha)$, a superposition of plane waves radiating in all directions from the aperture at $x = 0$ is, for $x \geq 0$,

$$E_y(x, y) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F\left(k_y\right) \exp\left[-j\left(k_x x + k_y y\right)\right] dk_y \ .$$

Setting $x = 0$, an inverse Fourier transform relation exists between the aperture distribution $E_y(0, y)$ and $F\left(k_y\right)$, and thus

$$F\left(k_y\right) = \int_{-\infty}^{\infty} E_y(0, y) \exp\left(jk_y y\right) dy \ .$$

The quantity $F\left(k_y\right)$ is called the plane wave spectrum of angular spectrum of the field in the aperture. By a concise development, [85] shows that, for a distant field point in the polar coordinates (r, θ), the vector field is

$$\mathbf{E}(r, \theta) = \hat{\theta} \frac{\pi}{\lambda} F(k \sin \theta) H_0^{(2)}(kr)$$

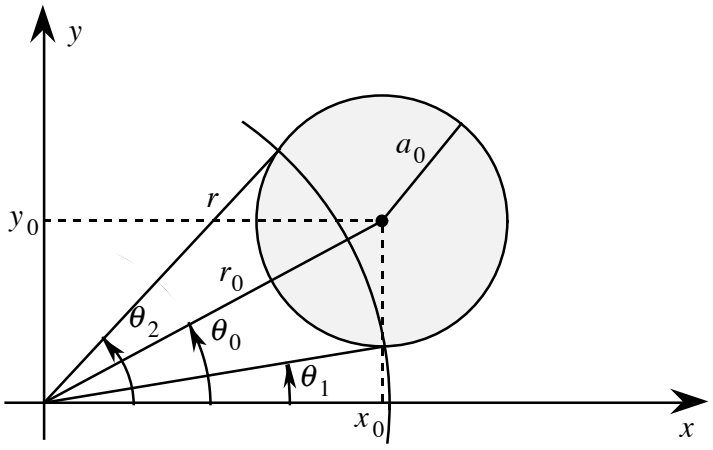


Fig. 4. 3. Geometry for finding the circular-arc projections of a top hat function in the special case of the radar being at the origin of the ground patch.

where $\hat{\theta}$ is a unit vector in the tangential direction and $H_0^{(2)}(kr)$ is the zero-order Hankel function of the second kind. By using an asymptotic form of the Hankel function for $kr \gg 1$, this reduces to

$$\mathbf{E}(r,\theta) = \hat{\theta}\,\frac{\exp\left[-j(kr-\pi/4)\right]}{\sqrt{r\lambda}}\,F(k\sin\theta) \tag{4. 4}$$

which represents cylindrical wavefronts or contours of equal phase, attenuating as $1/\sqrt{r}$ and having an angular dependence according to $F(k\sin\theta)$. The function $F(k\sin\theta)$ is also interpreted as the antenna far-field pattern.

The analogous result for the three-dimensional case has spherical wavefronts, an angular spectrum depending on two polar angles, and an attenuation proportional to $1/r$.

The value of having the salient features of the far field stated here is that algorithms can be developed to deal specifically with them, and if simplifications need to be made, then it is done knowledgeably.

4.2 Calculation of Circular-Arc Projections

This section will deal with the calculation of the return signal when the far field (4. 4) is appropriate and the ground patch reflectivity is the test function of Fig. 3. 1 and Table 3.1. The result is a generalization of (3. 1).

The geometry to begin the calculation is shown in Fig. 4. 3. A top hat of radius a_0 at a distance r_0 and angle θ_0 from the origin of the ground patch is shown. To simplify the situation, two important features of (4. 4) will be left out. These are the $1/r$ attenuation and the antenna pattern. It will be seen that correction for the range attenuation is trivial, and the correction for the antenna pattern will be discussed separately in Chapter 6. With this stated, temporarily let the radar be at

the origin, that is, $R = 0$. Owing to the flatness of the top hat, the problem is simply to find the length of the arc as it intersects the top hat and multiply by its height, here taken to be one (a "unit" top hat). With polar coordinates (r, θ), the circle defining the boundary of the top hat can be written

$$r^2 - 2r_0 r\cos(\theta - \theta_0) + r_0^2 - a_0^2 = 0.$$

Here, r is not only the radial coordinate but also the distance from the radar to the wavefront at which the projection value is desired. Solving this for θ gives

$$\theta = \text{Cos}^{-1}\left(\frac{a_0^2 - r_0^2 - r^2}{-2r_0 r}\right) + \theta_0$$

where $0 \le \text{Cos}^{-1}\,\phi \le \pi$. The angles θ_1 and θ_2 in Fig. 4. 3 are to be found. They are

$$\theta_2 = \theta_0 + \text{Cos}^{-1}\left(\frac{r_0^2 + r^2 - a_0^2}{2r_0 r}\right)$$

and

$$\theta_1 = \theta_0 - \text{Cos}^{-1}\left(\frac{r_0^2 + r^2 - a_0^2}{2r_0 r}\right). \tag{4. 5}$$

The desired line integral is then

$$2r\text{Cos}^{-1}\left(\frac{r_0^2 + r^2 - a_0^2}{2r_0 r}\right).$$

This result needs to be adjusted to allow for the radar being at some arbitrary location, (X, Y), as in Fig. 4. 4. Since the effect is a translation of the origin relative to the top hat, this requires a new value only for r_0; θ_0 does not appear. Let the new quantity be $\hat{r}_0$, so that the circular-arc projection can be expressed as

$$f_c(p, R, \Theta) = \begin{cases} 2\,r\,\text{Cos}^{-1}\left(\dfrac{\hat{r}_0^2 + r^2 - a_0^2}{2r_0 r}\right) \text{ if } -1 \le \dfrac{\hat{r}_0^2 + r^2 - a_0^2}{2r_0 r} \le 1 \\ 0 \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{4. 6}$$

where

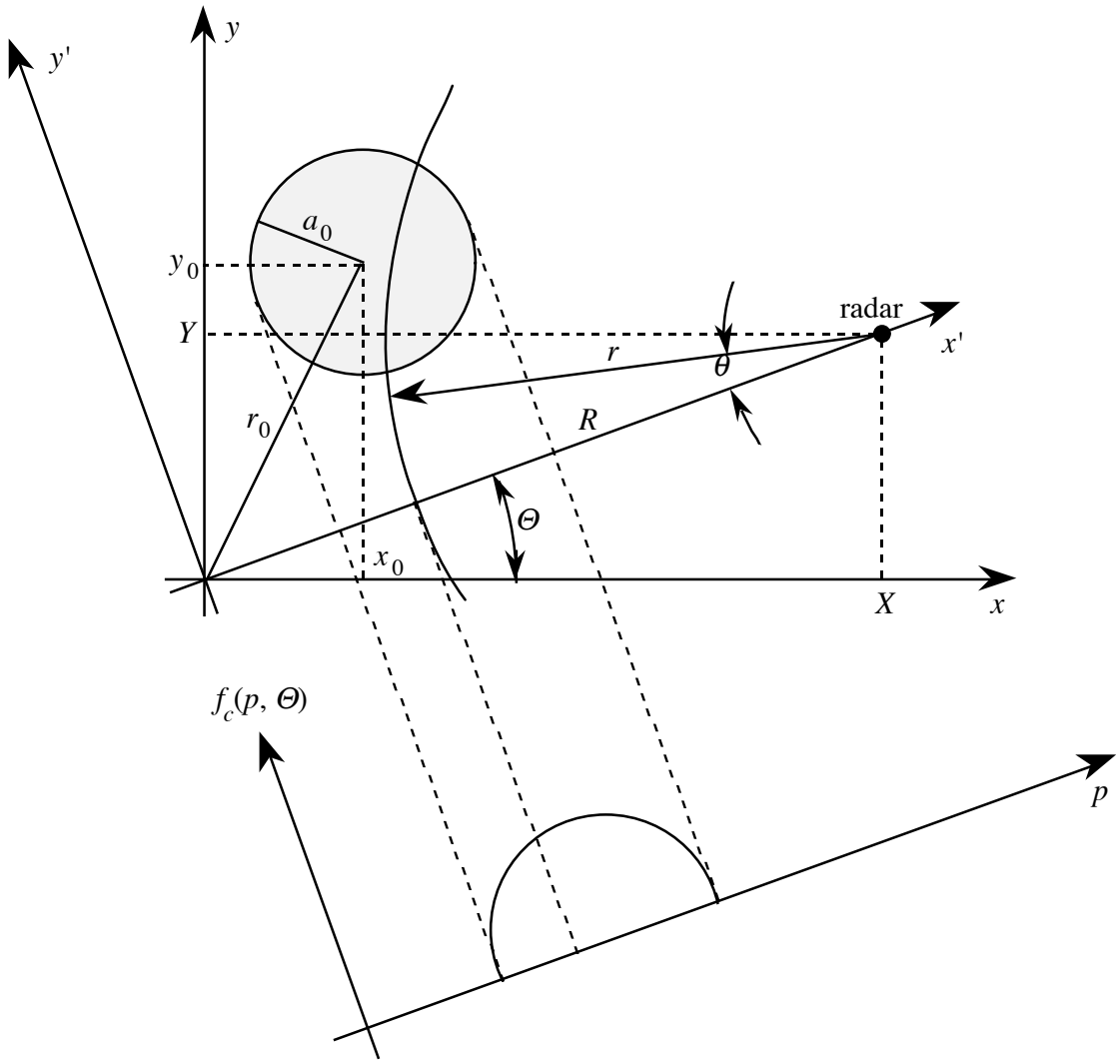


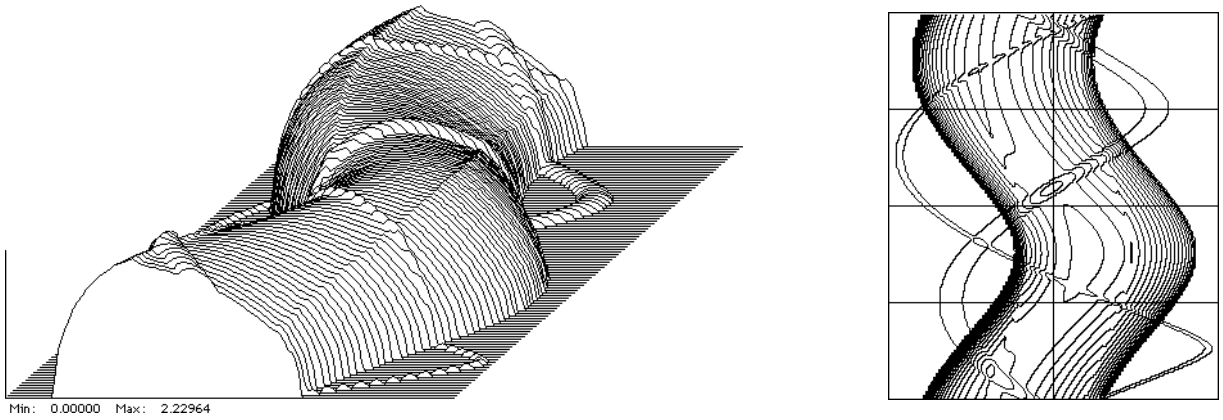
Fig. 4. 4. General geometry for finding the circular-arc projections of a top hat function, to be used in later simulations.

$$\hat{r}_0 = \left[(X - x_0)^2 + (Y - y_0)^2 \right]^{1/2}$$

and

$$p = R - r$$

which is used to maintain consistency with the p-variable of (3. 1). As before, p and x' are identical, although p is usually used in reference to the projection and x' is normally used to refer to the rotated coordinate system. The "otherwise" clause in (4. 6) is for the case when the circular arc does not intersect the top hat. The notation $f_c(p, R, \Theta)$ is intended to indicate the full functional



(a) (b)

Fig. 4. 5. Circular-arc projections of the test function. The radar traveled a circular path at a distance of 72 pixels from the center of the test function. (a) Perspective plot. (b) Contour plot.

dependence of the projections on the radar location.

The circular-arc projections of the test function are shown in Fig. 4. 5. The radar trajectory here was a circle centered on the origin of the ground patch of radius 72 pixels. Compared to a single straight-line projection of Fig. 3. 3, one of these appears approximately sheared in the horizontal direction. As a collection, there is some height modulation on the circular-arc projections and small features do not trace sinusoidal contours. The width at the base of the projection of a single top hat is the same as straight-line projections. A close study of these two figures may be worthwhile.

The comment in the above paragraph about projections of small features will be elaborated, as this will be found to be useful later. For straight-line projections, the transform of an impulse at (x_0, y_0) or polar coordinates (r_0, θ_0) can be shown to be

$$\mathsf{R}\{\delta(x - x_0, y - y_0)\} = \delta[p - r_0 \cos(\Theta - \theta_0)] \tag{4. 7}$$

where the operator $\mathsf{R}\{\bullet\}$ indicates the taking of straight-line projections, or the Radon transform, as before. Using the notation $\mathsf{C}\{\bullet\}$, the circular-arc projections of the same thing are

$$\mathsf{C}\{\delta(x - x_0, y - y_0)\} = \delta\left[p - R + \sqrt{R^2 + r_0^2 - 2Rr_0 \cos(\Theta - \theta_0)}\right]. \tag{4. 8}$$

The result

$$\lim_{R \to \infty} \mathsf{C}\{\delta(x - x_0, y - y_0)\} = \mathsf{R}\{\delta(x - x_0, y - y_0)\}$$

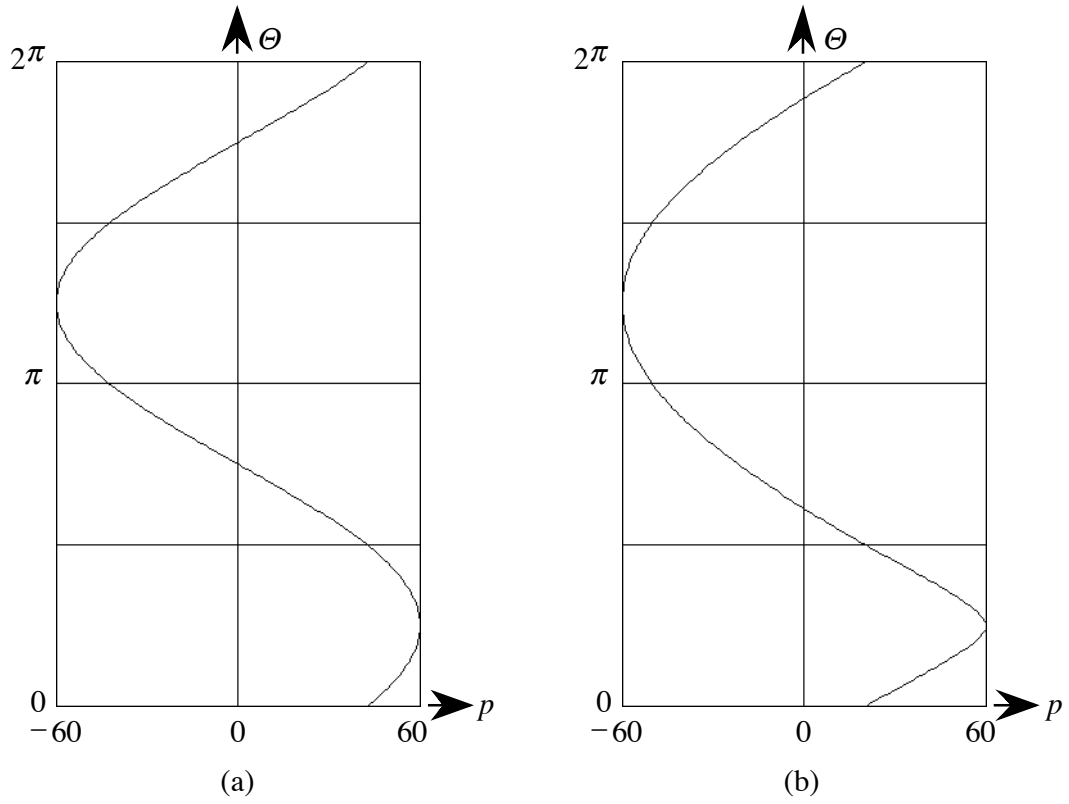


Fig. 4. 6. Contour plots of the projections of an impulse at $(r_0, \theta_0) = (60, 45°)$. (a) Straight-line projections. (b) Circular-arc projections with the radar at $R = 72$ pixels.

is easily shown. "Contour" plots of (4. 7) and (4. 8) are shown as Fig. 4. 6 (a) and Fig. 4. 6 (b) respectively — the independent variable p is horizontal here in order to be compatible with other contour plots of projections. The parameters for these plots are $(r_0, \theta_0) = (60, 45°)$ and $R = 72$. Equation (4. 8) is seen to be a distortion of (4. 7).

Finally, the circular-arc projections can be defined concisely in a form that is analogous to (2. 3) by expressing the line integral as

$$f_c(p, R, \xi) = \int f(\mathbf{x})\, \delta\left[(x - X)^2 + (y - Y)^2 - (R - p)^2\right] d\mathbf{x}. \tag{4. 9}$$

The argument of the δ function is a circle centered on the radar with radius $R - p$. The integration is understood to be restricted to arcs over the ground patch. It will be useful in the next section to simplify the notation in (4. 9). Letting

$$q(\mathbf{x}) = R - \left(R^2 + x^2 + y^2 - 2Rx\cos\phi - 2Ry\sin\phi\right)^{1/2}, \tag{4. 10}$$

(4. 9) can be written

$$f_c(p, R, \xi) = \int f(\mathbf{x})\, \delta[p - q(\mathbf{x})]\, d\mathbf{x} \ . \tag{4. 11}$$

4.3 Modified Convolution-Backprojection Image Reconstruction

This section presents a reconstruction algorithm which is appropriate for data collected along circular-arc projections. Three different interpretations are given. It will be seen that a simple modification to the standard convolution backprojection is sufficient to solve the problem. Computational aspects are addressed, and means for speeding up the computations by exploiting symmetry are examined. Compensation for the range factor $1/r$ can be incorporated as well.

The first presentation will be in terms of mappings on the image reconstruction plane.[8] While this approach may be slightly cumbersome conceptually, it seems to be best suited as an aid for writing computer programs. The central idea is to distort the reconstruction plane in such a way that convolution-backprojection can be used. Let (x, y) be the coordinates of the reconstruction plane, and let (x', y') be coordinates that are rotated by Θ, where the radar is at polar coordinates (R, Θ) in the (x, y)-plane. The geometry so far is completely analogous to the data collection geometry. The transformation from (x, y) to (x', y') is

$$\begin{aligned} x' &= x\cos(\Theta) + y\sin(\Theta) \\ y' &= -x\sin(\Theta) + y\cos(\Theta) \ . \end{aligned} \tag{4. 12}$$

The second mapping is from (x', y') to (x'', y''). Its purpose is to "straighten" the circular-arc projection paths, which are now rotated into a standard position with the radar on the x'-axis. A mapping is desired which preserves distance in both coordinates, i.e., the Jacobian is one. Such a mapping is given by (reference to Fig. 4. 4 may be helpful)

$$\begin{aligned} y'' &= r\theta \\ x'' &= R - r \end{aligned} \tag{4. 13}$$

where

$$\begin{aligned} r &= \sqrt{(x' - R)^2 + y'^2} \\ \theta &= \sin^{-1}\left(\frac{y'}{r}\right). \end{aligned} \tag{4. 14}$$

[8] Here, as in other places, the distinction between the image (ground patch) plane and the reconstruction plane will not be made explicit. It is assumed that the reader will be able to discern the difference. At times, however, since the reconstruction process in ways mimics the data collection process, it may not be necessary to make the distinction. This is not to diminish the conceptual importance of the difference.

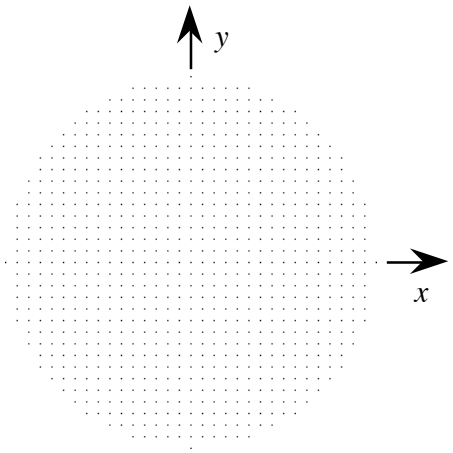


Fig. 4. 7. Undistorted image array before application of mappings to reconstruct from circular-arc data.

The Jacobian of the combined mappings is one. Note that the position of the radar, (R, Θ), has not been restricted in any way, and that the mappings change for each new pulse transmission, or radar position.

With these mappings, the algorithm can be motivated. If the *ground patch* could have been distorted according to these mappings, then *straight-line* projections would have yielded exactly the same data as circular-arc projections through the undistorted ground patch. Upon reconstruction, then, the mappings are applied for one value of Θ, the filtered projection is backprojected along straight paths, and the inverse mappings are applied. This is repeated for all projections, with the individual backprojections accumulated to form the final image. (The rotation stage, of course, is inherent in convolution backprojection.)

A grid of pixels is shown in Fig. 4. 7 before the mappings are applied. In this example, there are 33 by 33 pixels, restricted to a circular ground patch of radius 16, so that every fourth pixel in the actual reconstructions which follow is represented. In Fig. 4. 8 is shown what happens to this grid under the influence of the mappings for 18° increments over 90° of look angle variation. In this scale, the radar is at a distance of 18 pixels, which is also one fourth of the value of 72 which is used later for the reconstructions. It is over these points that the backprojection is executed normally, parallel to the y"-axis, before the inverse mappings are applied. One may observe that the mapped points for the 90° case are identical to the 0° case, not just a 90°-rotated version of it. Also, one may notice that the 36° and the 54° cases are mirror images of one another. These observations indicate that there is an eight-fold symmetry in the mappings. This will be explored later and will be found to be helpful in increasing the efficiency of the reconstruction algorithm.

The second interpretation of the algorithm, apparent from the first one, is that the backprojections are simply made along the same circular arcs from which the data were collected, the former

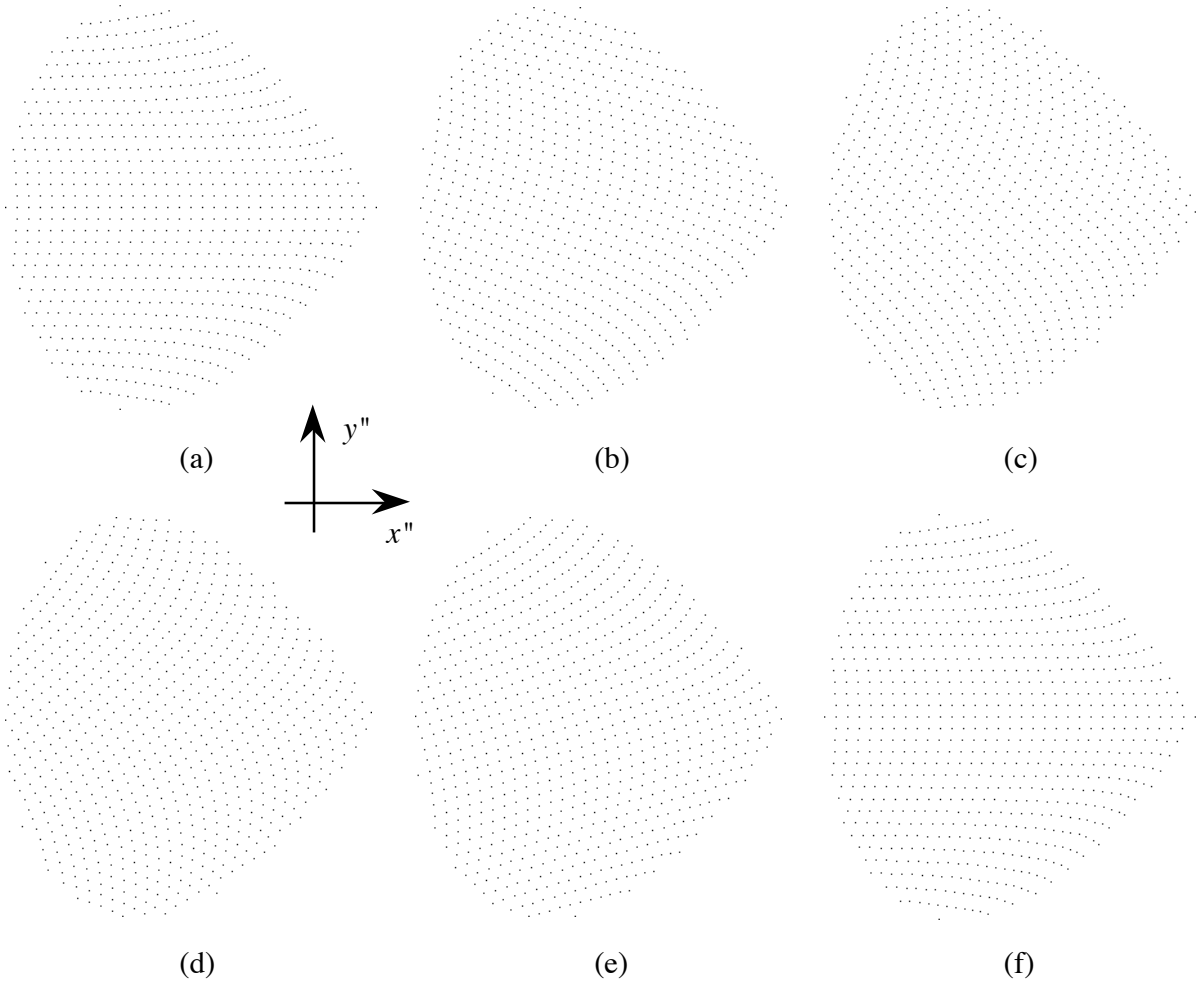
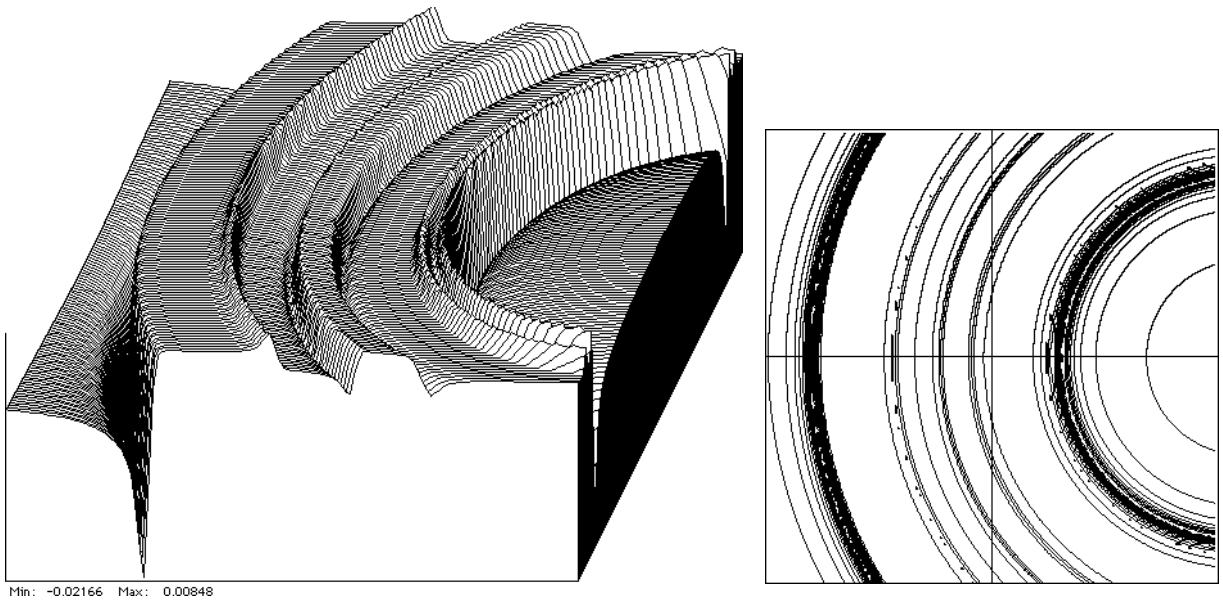


Fig. 4. 8. The image array of Fig. 4. 7 distorted by application of the mappings (4. 12) and (4. 13). (a) $\Theta = 0°$. (b) $\Theta = 18°$. (c) $\Theta = 36°$. (d) $\Theta = 54°$. (e) $\Theta = 72°$. (f) $\Theta = 90°$.

in the reconstruction plane, the later in the image plane. Discounting the convolution stage, this is appealing since the data are "put back where they came from" for each look angle, as best as can be determined given that there is the integral involved in the collection process. The operation of backprojecting along circular arcs is evident in Fig. 4. 9, which is the partial image of the test function, the reconstruction having been stopped after one backprojection, that backprojection being the one taken from $R = 72$, $\Theta = 0$.

This is a convenient place to formalize the reconstruction algorithm somewhat by developing a reconstruction formula in integral form. Notation from [22] will be used and modified. So far, "backprojection" has been used mostly to mean the process or result of converting a single filtered projection into a two-dimensional function by smearing it along some path, either linear or circular. At least once, in (2. 6), the term was used to refer to the process or result of smearing all of the filtered projections along different paths and summing. In [22], the first of these is called



(a) (b)

Fig. 4. 9. Backprojection of a single filtered circular-arc projection of the test function. Original projection angle was zero degrees, radar distance was 72 pixels. (a) Perspective plot. (b) Contour plot.

ϕ-*backprojection* or *natural transpose* and denoted in operator form for a function $h(p)$ as

$$\mathsf{B}_{\phi} h(p) = h(x \cos \phi + y \sin \phi)$$

which is seen to be a function of two variables, x and y, with ϕ as a parameter. The normal form for a line which intersects the x'-axis a distance p from the origin is

$$p = x \cos \phi + y \sin \phi = \mathbf{x} \bullet \xi$$

where $\mathbf{x} = (x, y)$ and $\xi = (\cos \phi, \sin \phi)$. The ϕ-backprojection can then be written

$$\mathsf{B}_{\phi} h(p) = h(\mathbf{x} \bullet \xi) \ .$$

The arguments in the above in relation to (2. 3) are to be noted.

A similar development can be made for the circular-arc backprojection. When the argument of (4. 9) is solved for p, the *circular* ϕ-*backprojection*

$$\mathsf{B}_{\phi}^{\mathsf{C}} h(p) = h\left[R - \left(R^2 + x^2 + y^2 - 2Rx \cos \phi - 2Ry \sin \phi\right)^{1/2}\right]$$

is obtained. The same result could be obtained by computing the 0-backprojection in the (x'', y'') plane,

$$\mathsf{B}_0\, h(p) \;=\; h(x'')$$

followed by mapping back into the (x', y') plane using the inverse mapping of (4. 13) and (4. 14), and from there back to the (x, y) plane using the inverse mapping of (4. 12). Since the functions which are circularly backprojected are the filtered projections instead of the generic $h(p)$, the entire image reconstruction process for data from circular-arc projections can be expressed in continuous-variable notation as

$$g(x, y) \;=\; \int_0^{2\pi} \overline{g}_c\left[R - \left(R^2 + x^2 + y^2 - 2Rx\cos\,\Theta \;- 2Ry\sin\Theta \right)^{1/2},\; \Theta \right] d\Theta \tag{4. 15}$$

where

$$\overline{g}_c(p, \Theta) \;=\; g_c(p, \Theta) * k(p)$$

$$= \frac{\partial}{\partial p}\left[\frac{1}{p} * g_c(p, \Theta) \right]$$

and where Θ has replaced the generic ϕ.

The justification for backprojecting along circular arcs can be made more rigorous. Here, it will be shown that circular-arc backprojection (with summation) is the adjoint operator of the circular-arc projection operator with respect to appropriate inner products on the two spaces.. This step is necessary to find the minimum-norm solution to the reconstruction problem. The proof is in the manner of Deans [22]. The circular-arc projection operator C is a bounded linear operator mapping images to collections of projections. Let f and g be images, and let f_c and g_c be their projections, i.e., $f_c = \mathsf{C}f$. Define an inner product in image space as

$$\langle f, g \rangle \;=\; \int f(\mathbf{x})\, g(\mathbf{x})\, d\mathbf{x}$$

and define an inner product in projection space as

$$[f_c, g_c] \;=\; \int_0^{2\pi} \int_{-\infty}^{\infty} f_c(p, R, \theta)\, g_c(p, R, \theta)\, dp\, d\theta \;.$$

Define the *circular-arc backprojection* of some function ψ as

$$\mathsf{B}^{\mathsf{C}}\psi(p, R, \theta) \;=\; \hat{\psi}(\mathbf{x}) \;=\; \int_0^{2\pi} \psi\left[q(\mathbf{x}), R, \theta \right] d\theta \;,$$

with $q(\mathbf{x})$ as in (4. 10). The adjoint $\mathsf{C}^\dagger$ is the operator which satisfies

$$\left\langle g, \mathsf{C}^\dagger f_c \right\rangle = \left[\mathsf{C}g, f_c\right] .$$

To show that B^C is the adjoint $\mathsf{C}^\dagger$, find

$$\left\langle g, \mathsf{B}^\mathsf{C} f_c \right\rangle = \int g(\mathbf{x}) \hat{f}_c(\mathbf{x})\, d\mathbf{x}$$

$$= \int g(\mathbf{x}) \int_0^{2\pi} f_c\left[q(\mathbf{x}), R, \theta\right] d\theta\, d\mathbf{x} .$$

Substituting the identity

$$f_c\left[q(\mathbf{x}), R, \theta\right] = \int_{-\infty}^{\infty} f_c(p, R, \theta)\, \delta\left[p - q(\mathbf{x})\right] dp$$

into the above gives

$$\left\langle g, \mathsf{B}^\mathsf{C} f_c \right\rangle = \int g(\mathbf{x}) \int_0^{2\pi} \int_{-\infty}^{\infty} f_c(p, R, \theta)\, \delta[p - q(\mathbf{x})]\, dp\, d\theta\, d\mathbf{x}$$

$$= \int_0^{2\pi} \int_{-\infty}^{\infty} f_c(p, R, \theta) \left\{ \int g(\mathbf{x})\, \delta[p - q(\mathbf{x})]\, d\mathbf{x} \right\} dp\, d\theta \ .$$

Using (4. 11), this reduces to

$$\left\langle g, \mathsf{B}^\mathsf{C} f_c \right\rangle = \int_0^{2\pi} \int_{-\infty}^{\infty} f_c(p, R, \theta)\, g_c(p, R, \theta)\, dp\, d\theta$$

$$= \left[\mathsf{C}g, f_c\right] .$$

This, along with the uniqueness of the adjoint [86], yields the desired result,

$$\mathsf{B}^\mathsf{C} = \mathsf{C}^\dagger .$$

The above result shows that circular-arc backprojection is in fact the correct operation if the minimum-norm solution is sought. However, it does not reveal *what* is to be backprojected. In the algorithm described here, filtered projections are backprojected, but this remains to be shown to be

correct. Medoff [51] examines the optimality of ordinary convolution backprojection under a different inner product and shows that filtering allows a minimum norm least squares solution. The extension of that work to encompass the present work remains to be done.

The third interpretation of the algorithm is assisted by (4. 15) and is also analogous to an interpretation of convolution backprojection known as *point-by-point* reconstruction [32]. In the present context, (4. 15) is still computed; the only difference is the order in which the computation takes place. The backprojection version of (4. 15) holds Θ constant while the integrand is evaluated for all of the desired values of x and y. After this is complete for all x and y, Θ is incremented and the integrand is again evaluated over all x and y, with the results summed to form the integral. The point-by-point process instead holds x and y constant while the integral over Θ is evaluated, yielding the reconstructed image value for that x and y. Then, new values of x and y are selected, and the process repeated. In this case, the integral is a line integral in the (p, Θ) plane along the path

$$p = R - \left(R^2 + x^2 + y^2 - 2Rx\cos\Theta - 2Ry\sin\Theta \right)^{1/2}$$

$$= q(\mathbf{x}).$$

This is seen to be exactly the same curve which describes the circular-arc projection of an impulse at (x, y), as shown in (4. 8). Therefore, Fig. 4. 6 (b) can also be used to indicate the path of the line integral. A similar development for the case of straight-line projections and a point-by-point interpretation of ordinary convolution backprojection shows a similar dual usage of Fig. 4. 6 (a). This interpretation provides more insight into why images reconstructed using plane-wave-based algorithms can have poor quality; the integration in the (p, Θ) plane is along the incorrect path. It can also be shown that the distortion of the curve in Fig. 4. 6 (b) diminishes relative to the curve in Fig. 4. 6 (a) when the distance of the point from the origin decreases, once again pointing up the problem of imaging "large" ground patches. This phenomenon will once more be demonstrated in the reconstructions which follow in this chapter.

Although computational aspects of the new algorithm will be discussed later in this chapter, it is worthwhile to mention here that the differences between the convolution backprojection method and the point-by-point reconstruction is also evident in a computer implementation, in which case discrete versions of the reconstruction integrals are implemented. Each program has three nested loops. In the convolution backprojection program, the outer loop ranges over Θ and the two inner loops select values for x and y. The innermost computation accumulates the integrand of (4. 15). In the point-by-point version, the only change is a reordering of the loops. The outer two loops are over x and y and the inner loop is over Θ. The data dependencies are different, too. For convolution backprojection, the next stage of image reconstruction can proceed as soon as the latest projection is available. Once that Θ -backprojection is complete, the filtered data can be discarded

and new processing can begin as soon as the next projection becomes available. For the point-by-point method, the entire set of filtered data must be available before reconstruction can begin, although the filtering can be done immediately. Therefore, although both algorithms result in exactly the same reconstruction, the point-by-point method results in a batch-mode latency and requires storage of all of the projection data at once. This latency can obviously be decreased by increasing processor throughput. Since the projections are known only on a discrete set, one advantage of the point-by-point method is that it can affect the way in which numerical algorithms are applied, such as integration and two-dimensional interpolation in the (p, Θ) plane. These prospects have not been explored.

With the concept of circular-arc projections with parameters R and Θ, one begins to wonder if there are any other applications in which projections of a function are taken along families of two-parameter curves in the plane. As the reader may have noticed, the answer is an immediate "yes." The point-by-point reconstruction method discussed above, for either the straight-line case or the circular-arc case, are just such projections. In the former case, the two parameters are the polar coordinates of the point being reconstructed: the radius of the point is the "amplitude" of a sinusoidal trace such as that in Fig. 4. 6 (a), while the angle is the "phase" of that trace. An analogous situation exists for the circular-arc case, with the parameters again being the polar coordinates of the point. So it is seen that there is a kind of duality between the data collection process and the reconstruction process.

So far, the $1/r$ attenuation in range has been ignored. Correcting for this effect is nearly trivial. All circular-arc projections are affected by this, including (4. 6). Inclusion of the attenuation would modify this by a factor of $1/r$. To correct for the effect, it is necessary only to multiply by $r = R - p$, and then the projections would be in a form expected by the reconstruction algorithm. In a real radar, the antenna is not in the plane of the ground patch and it has a radiation pattern which varies along its elevation axis. To the extent that this is independent of the antenna's azimuth axis, it effectively modifies the $1/r$ factor. To the same extent, this is just as easy to correct. Variation of antenna gain with azimuth is another matter, however, as seen in Chapter 6.

It should be stated that even though the Fourier domain interpretation and the concept of polar formatting [26], [65], do not apply here, it is perhaps best to consider this a polar-format algorithm since it includes convolution-backprojection as a limiting case, and since even short of that limiting case, it corrects for the same things as other, plane-wave, polar format algorithms.

4.4 Computational Considerations

The algorithm as presented so far appears to require more computation than it actually does, for three reasons. The first reason is that since the backprojection is parallel to the $y"$ axis, only the $x"$ part of the mappings needs to be computed. Consequently, only the first line of (4. 13) and the first line of (4. 14) need to be found. The second reason is more subtle. Only the mappings de-

scribed need to be computed explicitly; the inverse mappings, back to the (x, y)-plane, are free. Reference to Fig. 4. 7 and Fig. 4. 8 may help. The dots in Fig. 4. 7 represent pixel locations on a regular grid, such as is easily displayed on many image output devices. These pixels are represented internally as addresses in memory. The mapping is a mathematical operation which in an abstract sense takes the regular grid to the various distorted grids of Fig. 4. 8. Once this is done, there is no need to compute the inverse map pings, since all that is necessary is to "think" of the pixels as being back at their original locations; indeed, the computer and its display effectively do just that, automatically.

With these two adjustments to the computation, and with the loops ordered to implement backprojection (as opposed to point-by-point reconstruction), i.e., with the outer loop over look angle and the two inner loops over image pixels, and with a small amount of storage of reusable intermediate results, the computation can be optimized so that the part which must be in the innermost loop is actually quite modest, considering the original algorithm description. Since the computation which is in the inner loop dominates the overall running time on a single-CPU computer, including the convolution, this part is suggestive of the overall computational burden. (See [32] for a discussion of an implementation of plane-wave-based point-by-point reconstruction.) To be specific, when one-dimensional linear interpolation is used to get from sample values of the filtered projections to projected pixel locations, and for each execution of the inner loop, the algorithm requires seven additions, one multiplication, two squares, one square root, one rounding to minus infinity, plus loop control overhead. Of these, three additions, one multiplication, and one rounding to minus infinity are associated with the interpolation. The program listings in Appendix D require slight modification to be fully optimized; some speed was sacrificed for program clarity.

The third way of increasing the efficiency of the algorithm is based on an exploitation of symmetry that can be used in some circumstances. Inspection of the dot patterns of Fig. 4. 8 shows a great deal of symmetry. The patterns would repeat that sequence in every quadrant if they were continued over 360°. To investigate this symmetry more fully, suppose that both of (4. 12) were formally substituted into (4. 13). Then, there would be only three terms involved in the composite mapping, one of which is R, which is supposed for now to never change. The other two terms are

$$(x \cos \Theta + y \sin \Theta - R)^2$$

$$(-x \sin \Theta + y \cos \Theta)^2.$$

It is easily shown that these quantities are invariant with the following substitutions for x and y:

x	y	$-x$	$-y$	$-y$	x	y	$-x$
y	$-x$	$-y$	x	$-x$	$-y$	x	y

. (4. 16)

The first four of these (including the first one, which is trivial) show a four-quadrant symmetry. The last four show a mirror-symmetry with respect to each "arm" (positive or negative) of the x and y axes. Overall, these may be viewed either as a dual four-fold symmetry or an eight-fold symmetry—the latter will be used here. The above substitutions could be generated by finding all 12 permutations of the symbols $+x, -x, +y, -y$ and discarding the four that do not make sense here, such as x and x. A program which takes advantage of the eight-fold symmetry is included in Appendix D. Even though the program is more complex, its running time compared to a program which does not take advantage of symmetry is substantially reduced on at least one computer.

As a precondition for using the algorithm modified according to the eight-fold symmetry, it was mentioned above that R should be constant so that it is invariant under the substitutions (4. 16). This is a stronger condition than necessary, since any flight path which varies R in a manner consistent with (4. 16) is adequate. An example of such a flight path is a properly-positioned regular octagon. However, probably the most common situation which would work is the circular path, R a constant, such as turntable imaging.

A weaker, four-fold symmetry can be applied if only the first four substitutions of (4. 16) are used. Similar comments as in the eight-fold symmetry case apply with respect to restrictions on the variability of R over the flight path of the radar.

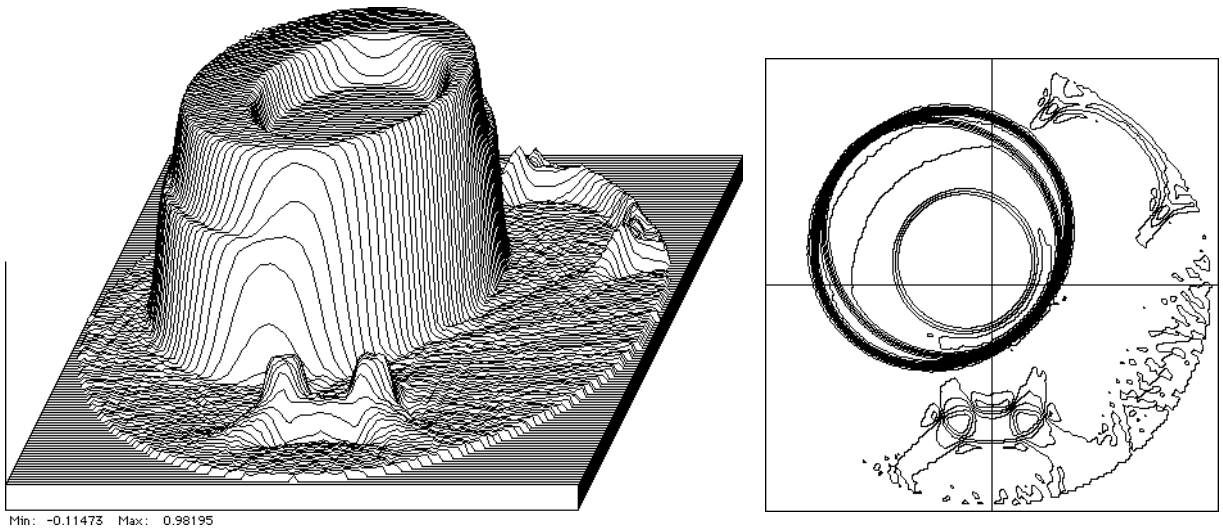
The mechanics of image reconstruction with the accelerated algorithms is as follows. In the case of four-fold symmetry, the backprojection angles begin from 0, $\pi/2$, π, $3\pi/2$, for example, and progress either clockwise or counterclockwise to the angle of the adjoining arm, so that the mappings need to be found for only one quadrant. In the case of eight-fold symmetry, the backprojection angles can have the same beginnings, but now the increments are both counterclockwise and clockwise from the axis arms, so that the mappings need to be found for only one-eighth of the circle.

A disadvantage of the accelerated algorithms is that the data dependency is changed. Even with the backprojection-style algorithm, there is still a batch-mode latency for the eight-fold symmetric algorithm amounting to the time it takes to collect very nearly 7/8 of the data. Of course, there are modifications which could trade off part or most of the latency for more storage of intermediate mappings.

Running times for the algorithms of this chapter are given at the end of the next section.

4.5 Simulations

The first reconstruction to be shown demonstrates what can happen when a plane-wave algorithm, here convolution backprojection, is used to reconstruct an image from data collected under rather severe conditions of wavefront curvature, Fig. 4. 10. The radar was at a constant distance of 72 pixels and circumnavigated the ground patch. There are many artifacts of the reconstruction, most of which might be described as focusing problems. The smallest top hat has been recon-



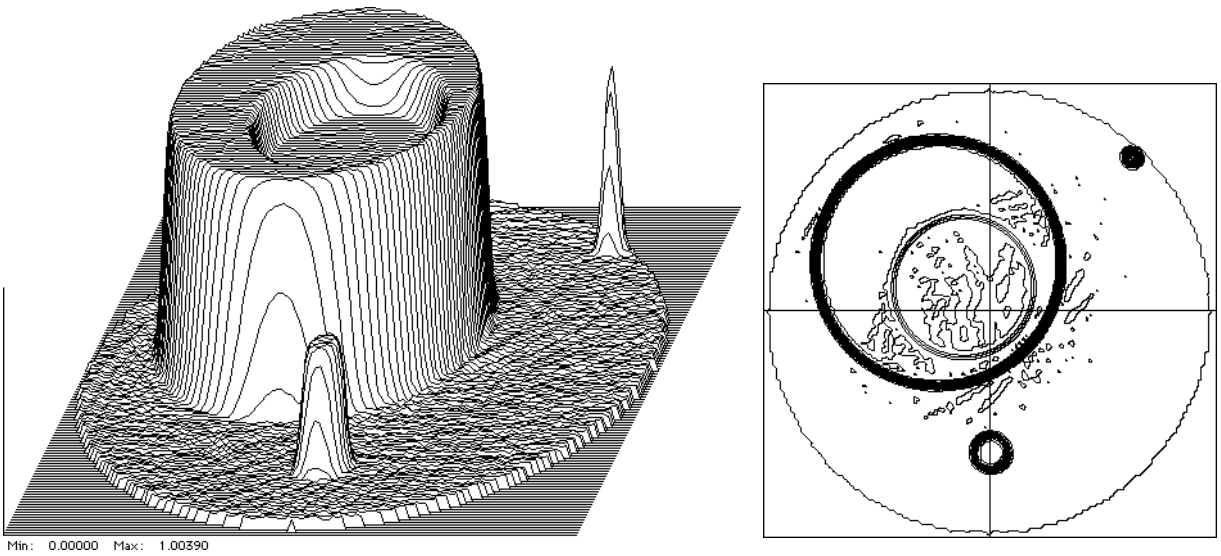
(a) (b)

Fig. 4. 10. Reconstruction from circular-arc projections using the unmodified convolution backprojection algorithm. Flight path was a circle of radius 72 pixels. (a) Perspective plot. (b) Contour plot.

structed very poorly, and the next smaller one fared little better. There is evidence that features at certain orientations are reconstructed with more damage than features at other orientations, as evidenced by the portions of the sides of the largest top hat which are made fairly steep. Apparently, all features near the origin are reconstructed relatively well. With a mental picture of how the backprojection process works and with the aid of the range error that is plotted in Fig. 4. 2, one can begin to see how this reconstruction was built. In the contour plot, it is interesting to notice that to the extent that the three major isolated features (the two smallest top hats separately and the two superimposed top hats together) do not affect one another during reconstruction, there seems to be some effect which makes them all have a symmetry about a suitably chosen radial line. These lines are approximately 45°, 135°, and 270° from the *x*-axis. It appears that all three major features are "facing" the origin.

The same circular arc data as were used in Fig. 4. 10 were also used in Fig. 4. 11, only here the "correct" algorithm, the modified convolution backprojection, was used. The reconstruction compares very favorably with Fig. 3. 6; in fact, Fig. 4. 11 is even better in some respects. Some visible differences in the two contour plots may be due to computed contours falling very near an area of very slight slope, such as in the vicinity of the origin. Also, there is some evidence of an overall shift in level, since the minimum which is reported in Fig. 4. 11 (a) is zero, but that is in the outlying area where the ground patch is not defined and where the image values are assigned a value of zero.

As promised, Fig. 4. 12 and Fig. 4. 13 show the effects of increasing the distance from the origin of a top hat more clearly than the previous reconstructions. The projections (not shown) are of

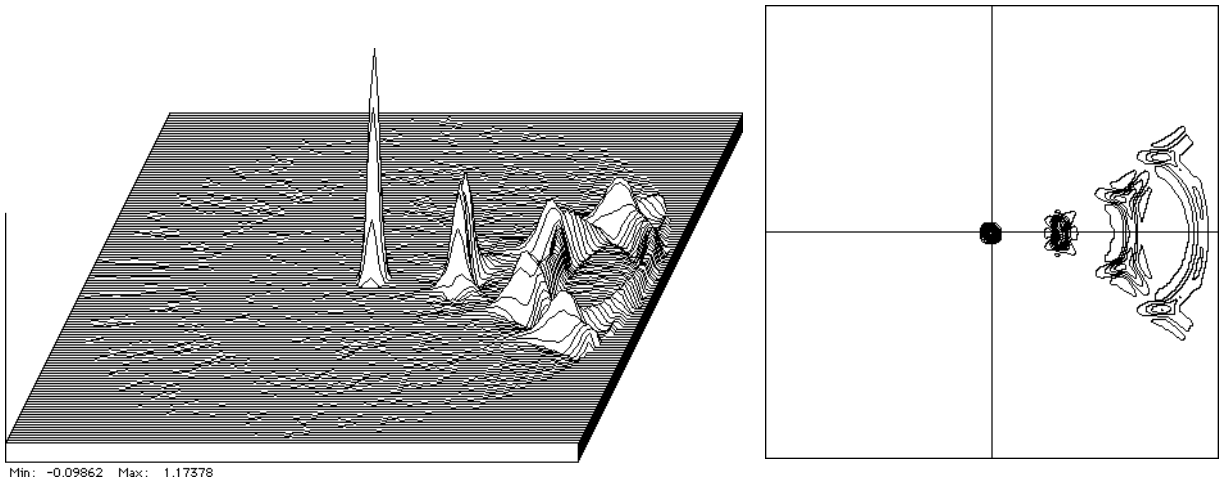


(a) (b)

Fig. 4. 11. Reconstructions from circular-arc projections using a modified convolution backprojection algorithm. The radar travelled a circular path of radius 72 pixels. (a) Perspective plot. (b) Contour plot.

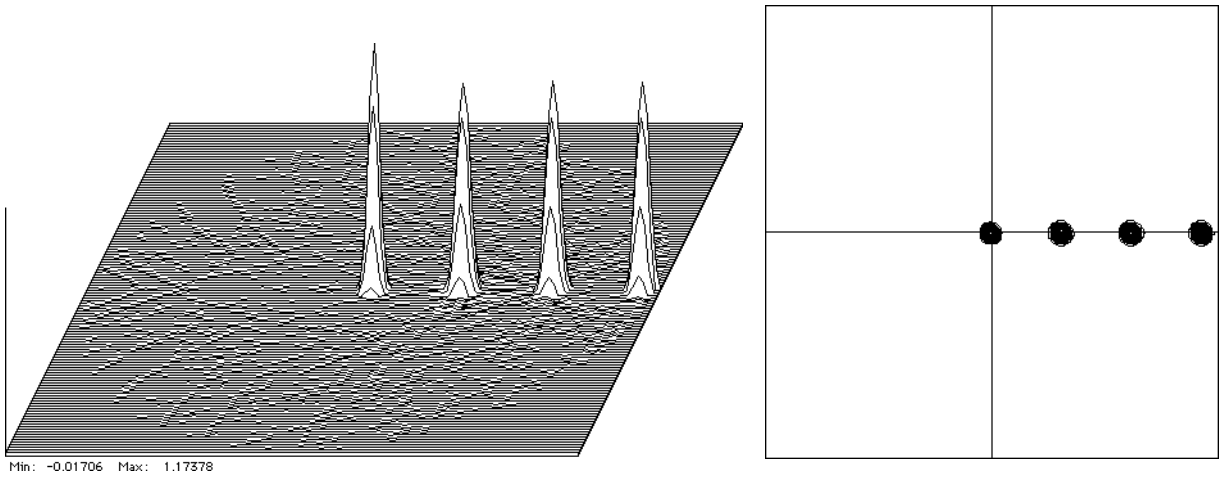
four identical unit top hats of radius two pixels, all on the x-axis, and at distances of zero, 20, 40, and 60 pixels from the origin. (It would be best to show each reconstructed top hat alone, but all are lumped into one reconstruction here to save space—even in the worst case, they apparently interfere little with one another.) A reconstruction using the plane-wave algorithm is shown in Fig. 4. 12. As predicted, the top hat at the center is reconstructed very well, with increasing focus problems with distance from the origin for the others. The height of the peaks decreases and the blurring becomes worse, being generally distributed along lines that are perpendicular to radial lines. As verification of the new algorithm, it was used to reconstruct the same function, the result being shown in Fig. 4. 13. All four top hats are reconstructed very well, with the main limitation on focus probably still due to the Hamming window. The three outer ones are virtually identical while the central one is slightly narrower. The source of the additional height of the central top hat is not known; careful measurements of the other three show that they are almost exactly unit height. It should be noted that the center pixel, with the projection and reconstruction parameters used here, is the only one which does not require interpolation from the filtered projections—perhaps there is a natural overshoot from top hats of this size and the interpolation process affects the other three to slightly "dull" them.

As a further comparison to a baseline reconstruction of Chapter 3, (Fig. 3. 8), Fig. 4. 14 is a reconstruction of the circular-arc projections of an impulse at the origin of the ground patch using the modified algorithm. These projections are the same as the straight-line projections of the same function. Both reconstructions are similar, with the newer one perhaps slightly better in the outlying pixels. The contour plot is again made using 500 levels in order to show up the outlying areas.



(a) (b)

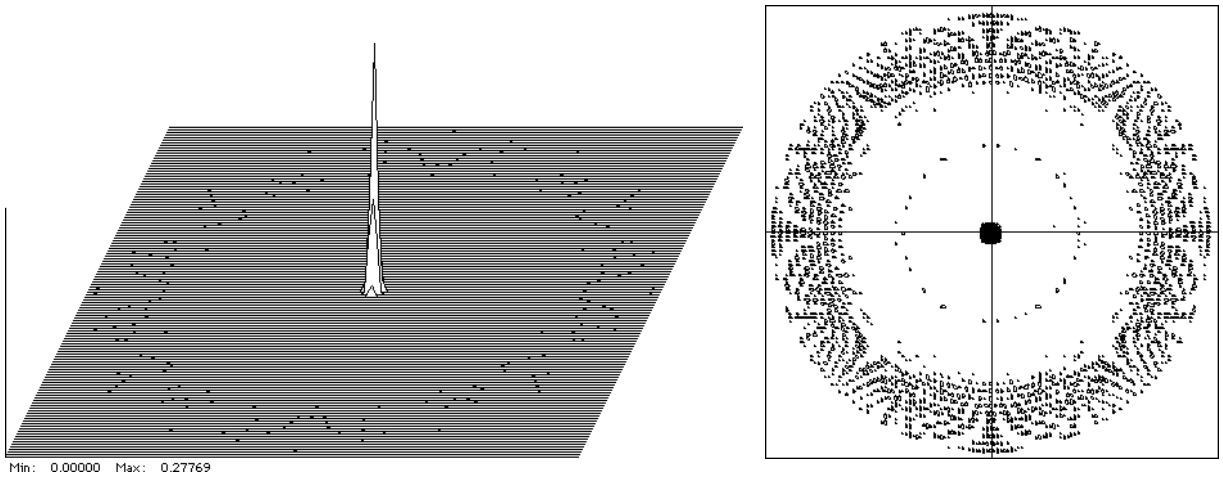
Fig. 4. 12. Reconstruction of four unit top hats using convolution backprojection to show worsening focus with increasing distance from the origin. (a) Perspective plot. (b) Contour plot.



(a) (b)

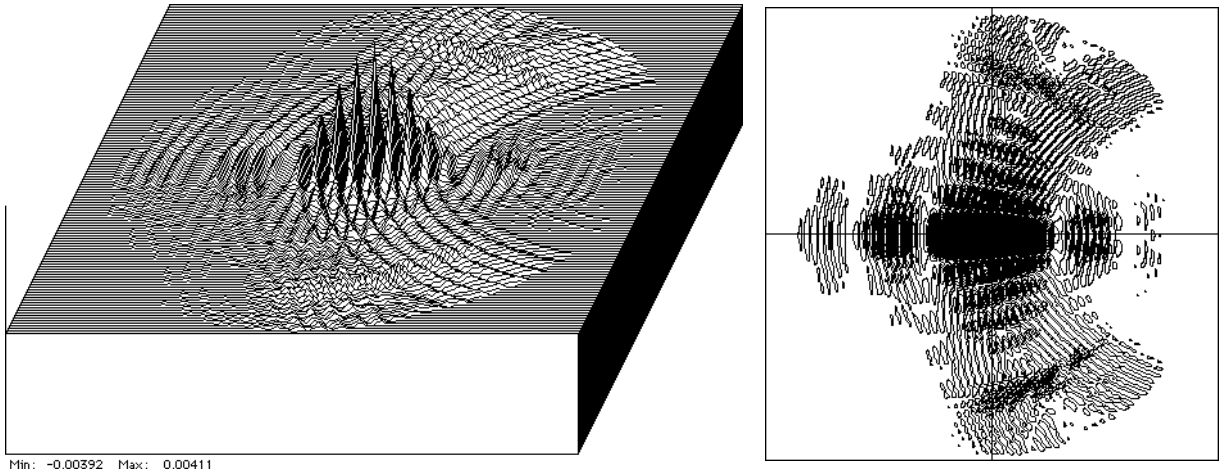
Fig. 4. 13. Reconstruction of four unit top hats using modified convolution backprojection to show constant focus with increasing distance from the origin. (a) Perspective plot. (b) Contour plot.

Shown in Fig. 4. 15 is the reconstruction which is analogous to Fig. 3. 9, that is, from bandlimited projection data from a centered impulse. Although the Fourier domain interpretation of bandlimiting does not hold here, there is no problem in describing the simulation; it was conducted in exactly the same way as was that for Fig. 3. 9. The look angle was restricted and the filtering was modified to simulate bandlimiting from a pulse of finite length. As in the earlier figure, down conversion was not done. Streak artifacts can be seen which are analogous to those in Fig. 3. 9. Otherwise, the two reconstructions have much of the same character.



(a) (b)

Fig. 4. 14. Reconstruction of an impulse from its circular-arc projections using a modified convolution-backprojection algorithm. (a) Perspective plot. (b) Contour plot (500 levels).



(a) (b)

Fig. 4. 15. Reconstruction of a central impulse from bandlimited data using the modified algorithm. (a) Perspective plot. (b) Contour plot (100 levels).

So far, all of the simulations in this chapter have had the radar traveling in a circular path of radius 72 pixels centered on the ground patch. The next two simulations will alter this pattern. First, the radar is made to travel a square trajectory, also centered on the ground patch, transmitting with a uniform PRF. Each side of the square is 144 pixels long. Shown in Fig. 4. 16 are the radar locations at the times of these transmissions and receptions, only with some left out for clarity. (This simulation requires that the number of projections be divisible by eight, so 200 are used.) The radar coordinates, rectangular or polar, are to be thought of as being functions of the cumulative distance which is measured from the initial radar position on the x-axis, that is, $(X(s), Y(s))$ or

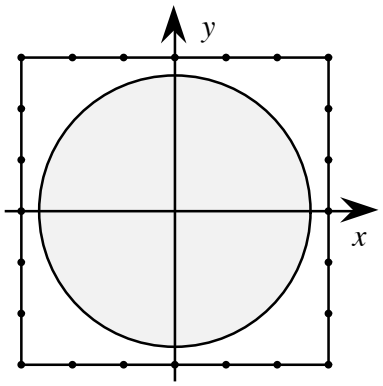


Fig. 4. 16. Trajectory of the radar for a particular simulation. Dots represent 24 of the 200 positions of the radar transmitting with a uniform PRF. Ground patch shown shaded, to scale.

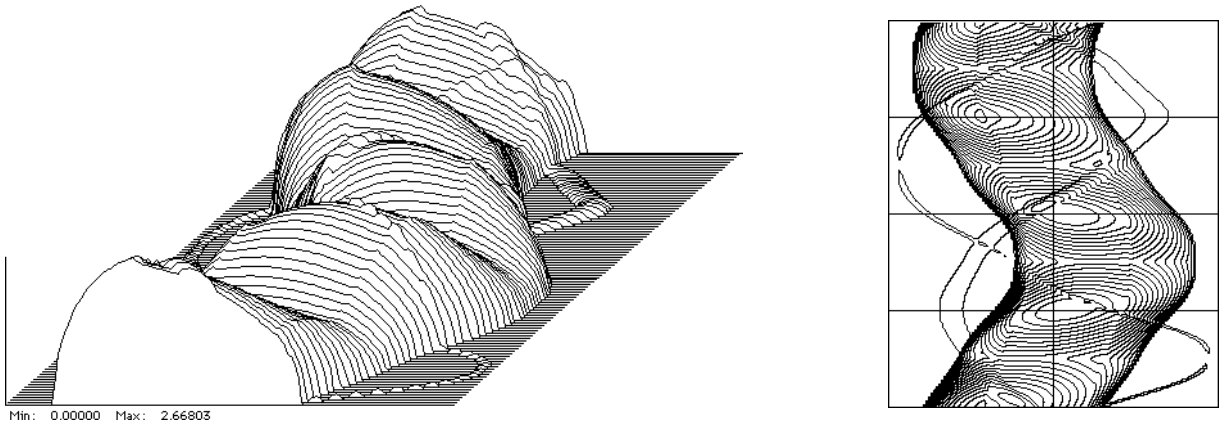
$(R(s), \Theta(s))$. Also of importance here is that since the increments in angle from pulse to pulse are not uniform, each projection is weighted according to its relative increment, i.e ., the $\Delta\Theta$ which is used to approximate $d\Theta$ in (4. 15) when (4. 15) is approximated as a finite sum must vary. The method used here is simply

$$\Delta\Theta_n = \frac{\Theta_{n+1} - \Theta_{n-1}}{2} \tag{4. 17}$$

where $n = 0, 1, \ldots N - 1$ is the pulse index and is counted modulo N in order to be consistent with Θ being modulo 2π, in order to account for indexing at the point of wrap-around.

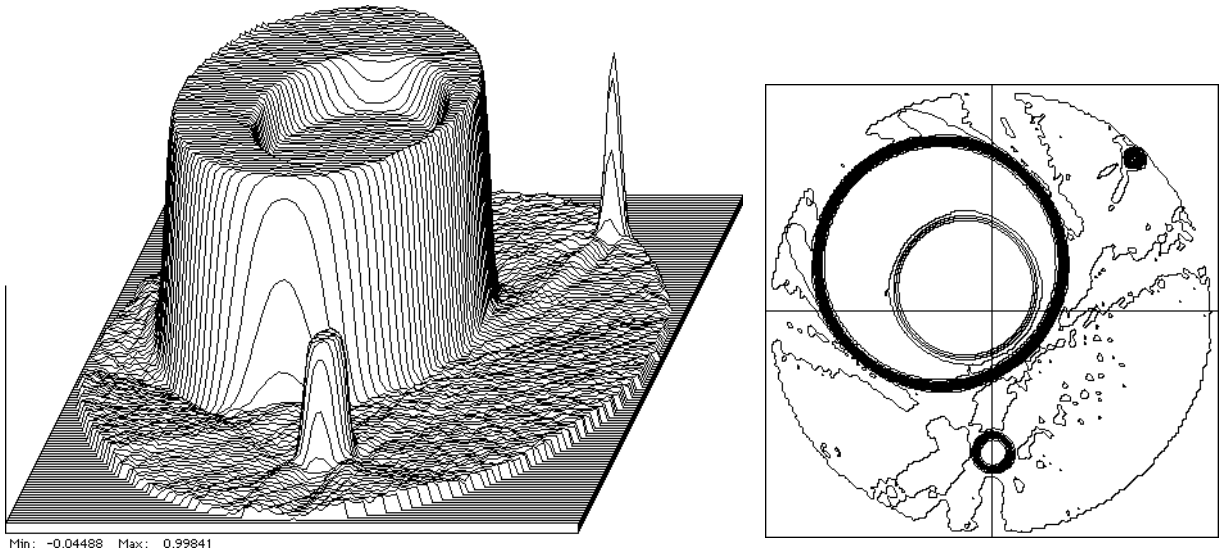
The projections which result from the sampling scheme of Fig. 4. 16 are shown, with the weighting of (4. 17), in Fig. 4. 17. Some of the height modulation is due to the weighting and some of it is due to the variability of R, as may be seen in (4. 6) through the dependence on R, X, and Y. The reference for p, $p = 0$, is for the arc which passes through the origin of the test function, regardless of the position of the radar. The interpretation of the coordinates for this figure is that the horizontal one is still p but the depth coordinate should be thought of as s.

The reconstruction algorithm proceeds just as before, only with information regarding the flight path that was taken during data collection. (Technically, the weighting of (4. 17) should be thought of as part of the reconstruction algorithm.) The circular backprojections are adjusted so that the center of the circles corresponds to the location of the radar, projection by projection. The resulting reconstruction is shown in Fig. 4. 18. While this reconstruction is good, the presence of low-level streak artifacts makes it inferior to Fig. 4. 11, for example. These artifacts might be due to secondary numerical effects such as the use of (4. 17), or there might be a more fundamental reason. It appears that with R varying, some parts of the reconstruction plane are being "covered" by the several backprojections more densely than others. This possibility has not been examined.



(a) (b)

Fig. 4. 17. Circular-arc projections for the radar trajectory of Fig. 4. 16, weighted according to non-uniform angular increments. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 4. 18. Reconstruction from the projections of Fig. 4. 17. (a) Perspective plot. (b) Contour plot.

Another possibility for projections collected from nonuniform angular increments is to interpolate the available projections to uniform increments, perhaps using knowledge of the flight path.

The final simulation has the radar on the trajectory shown in Fig. 4. 19. The pulses are transmitted such that the angular increments are uniform. Not shown are two samples at $(\infty, 0)$ and $(-\infty, 0)$. As always, the radar starts collecting data when on the positive x-axis and travels counterclockwise. The point of closest approach is 72 pixels. The projections are shown in Fig. 4. 20 and the reconstruction in Fig. 4. 21. Errors in the reconstruction invite comments similar to those made with respect to Fig. 4. 18 in terms of fine tuning the algorithm for this case.

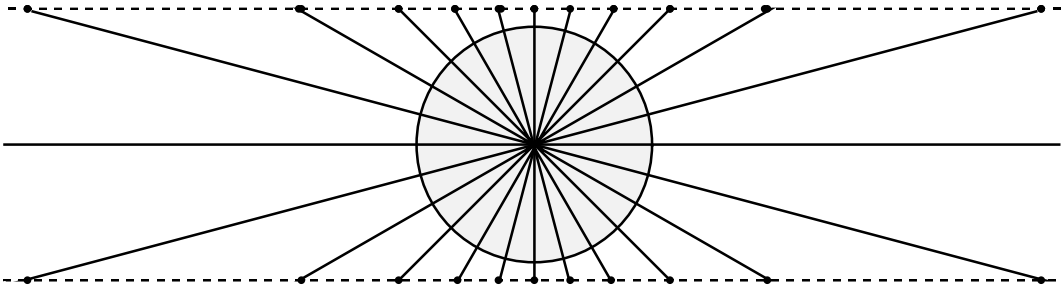
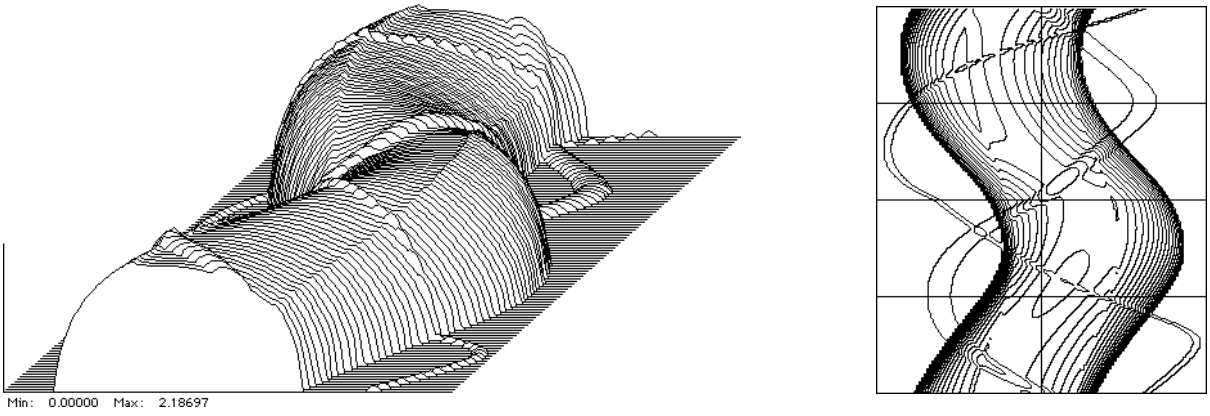
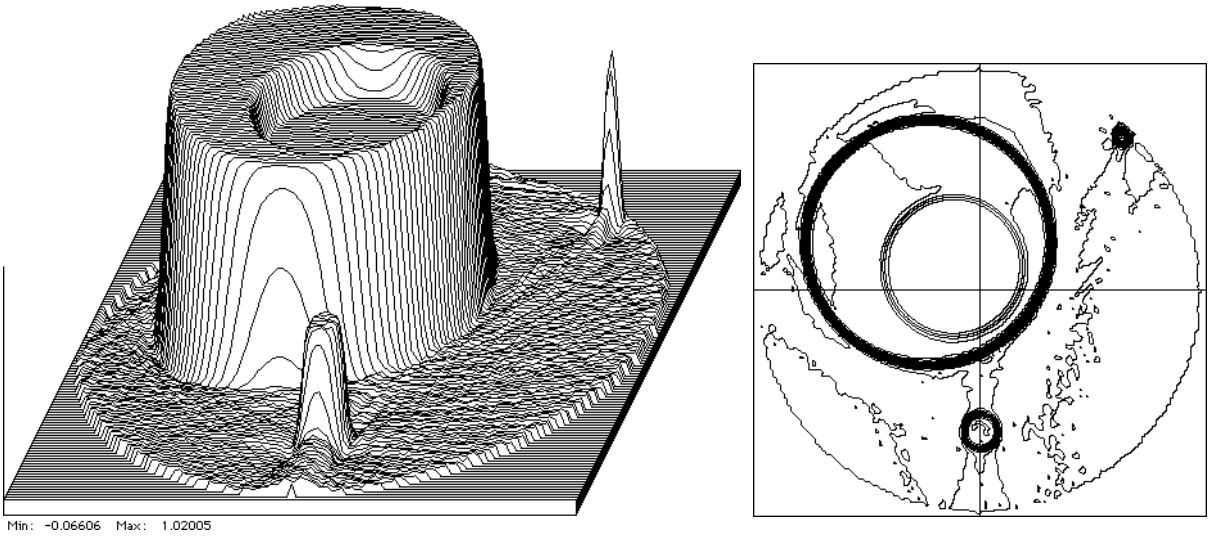


Fig. 4. 19. Trajectory of the radar for a particular simulation. Dots represent 22 of the positions of the radar transmitting with a nonuniform PRF. Not shown are dots at infinite distance on the x-axis. Ground patch shown shaded, to scale.



(a) (b)

Fig. 4. 20. Circular-arc projections for the radar trajectory of Fig. 4. 19. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 4. 21. Reconstruction from the projections of Fig. 4. 20. (a) Perspective plot. (b) Contour plot.

It is worthwhile to quote the running times of the programs used so far in the simulations. All times relate to the use of THINK's Lightspeed Pascal, published by Symantec Corporation, running on an Apple Macintosh II computer with 2 MB of random access memory and a 40 MB hard disk drive with an access time of 29 ms. This computer uses the Motorola 68020 microprocessor and the 68881 floating point unit performing 80-bit floating point arithmetic according to the IEEE 754 standard. Both processors are clocked at 15.6672 MHz. Projection data and reconstructed-image data are stored on disk in 32-bit floating point format. The program that is used for the baseline simulations of Chapter 3 is called CBP, for Convolution Back-Projection. It runs in 8.5 min. The most general program that works for wavefront curvature with monostatic SAR (as in this chapter) is called CACBP for Circular-Arc Convolution Back-Projection, and runs in 12.7 min. The version of this program that exploits eight-old symmetry, CACBP8, runs in 6.1 min. Code for these programs comprises part of Appendix D.

4.6 Quadratic Phase

The term *quadratic phase* is encountered rather frequently and has several different meanings in radar, especially SAR. (This is in addition to the term *quadratic phase error*.) This section will briefly discuss the relationship of the new algorithm to two uses of the term which could be confused because of occasionally casual and ambiguous usage in the literature.

The first occurrence of the term quadratic phase to be mentioned is in Equation (10) of [26] which, after conversion to the present notation, is

$$C_\theta(t) = \int_{-L}^{L} f_l(x', \theta) \exp\left(j\frac{4\alpha x'^2}{c^2}\right) \exp\left\{-j\frac{2}{c}[\omega_0 + 2\alpha(t - \tau_0)]x'\right\} dx' .$$

This is an expression for the return signal from a ground patch, the projections of which are $f_l(x', \theta)$ after processing by the receiver. A constant attenuation factor has been eliminated. The FM chirp rate is 2α, the round-trip delay from the radar to the center of the ground patch is τ_0, and the center frequency is $\omega_0/2\pi$ Hz. The part of the expression which is a quadratic phase,

$$\exp\left(j\frac{4\alpha x'^2}{c^2}\right),$$

is due to the choice of a chirp as the transmitted signal and the choice of receiver. [9] The presence of this term is not the result of any relevant approximation and will be present regardless of the

[9] The type of receiver assumed in [26] mixes the received signal with a delayed copy of the transmitted signal, followed by low-pass filtering. A detailed analysis, included as Appendix A, shows that this receiver approximates a matched filter receiver if the transmitted signal is a linear FM type of large time-bandwidth product and if the variation in the delays of the reflected signals is small compared to the transmitted signal duration.

shape of the wavefront of the transmitted signal. In [26], plane waves are assumed throughout the analysis, maintaining perfect consistency. Projections of the ground patch reflectivity are assumed to be along straight lines. However, if the object is to simply calculate the return signal (and not to devise a new algorithm), then the projections could be assumed to be from along circular arcs or any other path, and the quadratic phase term would remain, even if only a point target were assumed. This quadratic phase is a result of the time-bandwidth product of the transmitted chirp being finite; consequently, such a receiver cannot be a perfect Fourier processor, although it can come close. Interestingly, this quadratic phase term is exactly like the term in wave theory that is discarded when passing from the Fresnel region to the Fraunhofer region of radiating sources (see, for example, [87]), the radiation pattern in the latter region bearing a Fourier transform relationship to the source distribution. Also, a calculation of the Fourier transform of a chirp signal requires the use of the Fresnel integrals.

The second occurrence of quadratic phase is in relation to a geometric approximation which is pervasive in the literature [65], [44]. This approximation is alluded to in Chapter 1, and results when the exact distance from the radar to a point on the ground is written using Pythagoras' formula and simplified using a truncation of its binomial expansion. Almost always, two terms are retained, leading effectively to a locally plane wave approximation of a spherical wave. Occasionally, another term, the "quadratic" term, is added in an analysis of the effects of ignoring it. The comments of the previous paragraph with respect to Fresnel and Fraunhofer regions notwithstanding, this occurrence of quadratic phase is unrelated to the first occurrence. It is, however, closely related to textbook-like radiation effects, but the radar case of two-way propagation needs to be treated very carefully in determining under what conditions the far field situation is applicable. (It is easy to show, for example, that even if all points of the ground patch are in the far field of the radar antenna in transmit mode, it is almost certain that the radar antenna is *not* in the far field of the ground patch when it receives the reradiated energy. Further, it is easy to show that increasing the distance between the radar and the ground patch only worsens the situation. These results comprise Appendix B. The effect of this phenomenon on image quality is not known.)

The relationship of the new algorithm to these two kinds of quadratic phase is that the first kind is not affected at all and the second kind is completely corrected, along with all of the higher-order terms in the binomial expansion. The first kind is not affected because it is strictly a consequence of signal and receiver design which are beyond the scope of, and have no bearing on, the new algorithm. The second kind is corrected because it is the essence of wavefront curvature for which the algorithm is specifically designed to take into account.

CHAPTER 5

BISTATIC SAR

BISTATIC SAR shares many characteristics with monostatic SAR, but it also has many unique characteristics which make it a worthy subject of study on its own. In principle, monstatic SAR is a special case of bistatic SAR in which the same antenna is used for both transmission and reception, rather than using two spatially-separated antennas. However, to study monostatic SAR as a special case is to load the problem with superfluous facts which only require extra effort to discard as they are no longer needed. Also, in this dissertation, the study of the monostatic case was a learning exercise during which experience was gained relating to the process of modifying convolution backprojection under conditions of significant wavefront curvature. The experience gained from this step was valuable in extending the work to the bistatic case, and so the presentation here will follow the order in which the work was done. It is hoped that the reader will similarly benefit.

Given the framework of projectional sampling of the ground patch reflectivity, the effect of the plane wave and spherical wave assumptions with bistatic SAR is readily understood. Under the plane wave assumption, it is easy to show (with a judicious choice of coordinate system) that the projections which result are the same projections that would be obtained from a monostatic radar at an angle which would bisect the angle formed by the bistatic antennas (the bistatic angle). This statement depends on both of the bistatic antennas and the hypothetical monostatic antenna being a great enough distance from the ground patch that plane waves are an adequate description of the field over the ground patch. If a proof is undertaken, it is helpful to find the contours of equal times-of-flight from two very large antennas at a finite distance from the ground patch, each of which is phased to transmit and receive plane waves perpendicular to its length, and at the same angles as the actual bistatic antennas. A suitable reconstruction algorithm for this case is any Fourier algorithm which is based on variation in the bistatic angle bisector. Such variation may be obtained by motion of either or both of the antennas during data collection.

If either or both of the bistatic antennas approach the ground patch (the separating distance is no longer much greater than the size of the ground patch), then once again the spherical model for wave propagation should be used in order to design algorithms for high-quality image reconstruction. Since the speed of propagation is constant, time maps into distance so the contours of equal times-of-flight, using the “two pegs, string loop, and a pencil” construction, are easily seen to be a family of confocal ellipses. In three dimensions, these contours are a family of confocal prolate el-

lipsoids with the transmitting antenna at one focus and the receiving antenna at the other focus. Plane intersections through the antenna locations are the confocal ellipses. Therefore, the model which is used here for radar return is that of elliptical-arc projections through the ground patch. By elliptical-arc projection it is meant that the path of integration in forming the projection is part of an ellipse through the ground patch. The variation in projections is obtained by moving one or both of the antennas over a suitable region of space. In this chapter, it will be assumed that the bistatic system travels around the ground patch in some fashion; in Chapter 6 a proposal is made for a somewhat different kind of bistatic imaging, one which capitalizes on wavefront curvature and the speckle imaging phenomenon.

This chapter will examine two types of elliptical-arc projections: the first case applies when the propagation attenuation can be ignored and the second case includes it. The reason for separating the two cases is that, at least for the algorithms derived herein, correcting for propagation attenuation in bistatic SAR greatly expands the computation required over the algorithm which is uncorrected for this effect. This is in contrast to monostatic SAR in which correcting for the propagation attenuation is nearly trivial, as discussed in Chapter 4.

The presentation of material in this chapter is slightly different than in previous chapters. As before, the calculation of projections is discussed first. However, the algorithm development is closely tied to the simulations, requiring some iteration in the simulations to understand what is needed to adjust the algorithm, and so these steps are presented as one. This method is in lieu of a comprehensive theory of image reconstruction for the variety of situations studied. In order not to be overwhelmed with solving the whole problem at once, the unattenuated case is studied before the attenuated case, and a simplified trajectory for the radar antennas is used before passing on to more general trajectories.

5.1 Calculation of Elliptical-Arc Projections

The calculation of projections for bistatic SAR is more involved than in previous cases. Shown in Fig. 5. 1 is the geometry, the salient features of which include a new coordinate system called the bistatic coordinates which has the y_b-axis such that it is directed from the transmitter to the receiver, its origin is at rectangular coordinates (X, Y) and polar coordinates (R, Θ) in the x-y system, and it is tipped by an angle Θ_b from the x-axis. The transmitter and receiver are at coordinates (0, –F) and (0, F) in the bistatic system, respectively, and at (X_t, Y_t) and (X_r, Y_r) in the x-y system, respectively. The smaller circle represents a typical top hat of the test function at rectangular coordinates (x_0, y_0) and polar coordinates (r_0, θ_0). Angles θ_1 and θ_2 denote the points of intersection of an ellipse which has the transmitter and receiver at its foci with the edges of the top hat. The ellipse intersects the x_b-axis at A and the y_b-axis at B. The relationship between A, B, and F is

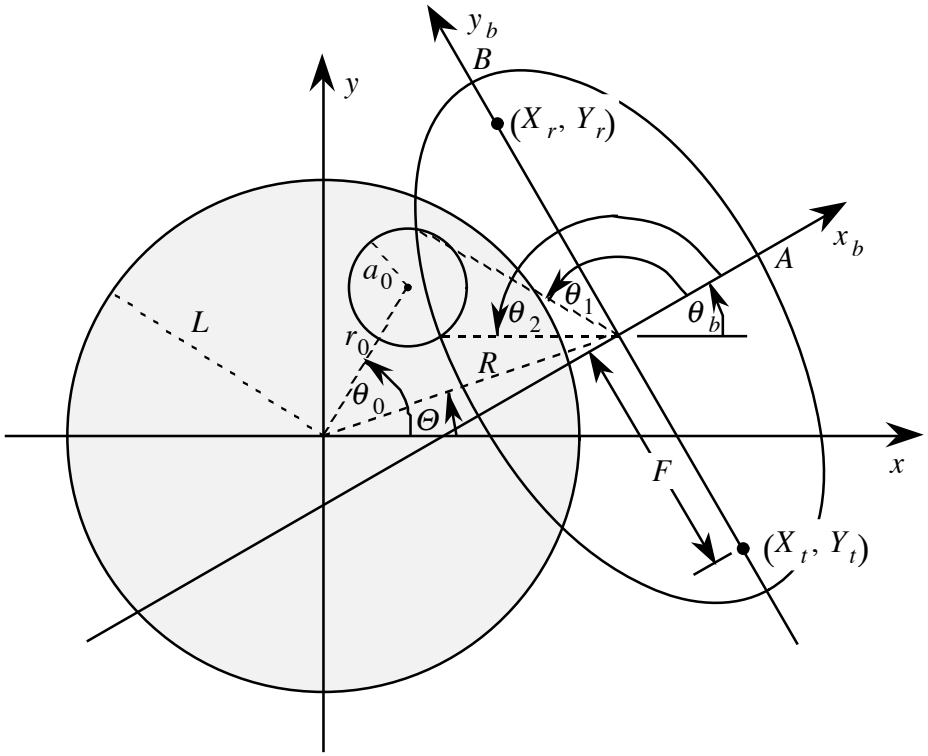


Fig. 5. 1. Geometry of bistatic SAR.

$$F = \sqrt{B^2 - A^2}. \tag{5. 1}$$

The sum of distances from any point on the ellipse to the foci is $2B$. This fact will be useful later in a discussion of sampling issues in bistatic SAR and in algorithm development.

5.1.1 Without propagation attenuation

The task of finding the elliptical-arc projections of a unit top hat is the same as finding the length of the arc between θ_1 and θ_2. The first step is to find these angles. It will be convenient to work in the bistatic system so that the ellipse is always in an "upright" position. Points in the *x-y* system, such as the boundary of a top hat, can be transformed into points in the bistatic system by the mapping

$$\begin{aligned} x_b &= (x - X)\cos\theta_b + (y - Y)\sin\theta_b \\ y_b &= -(x - X)\sin\theta_b + (y - Y)\cos\theta_b. \end{aligned} \tag{5. 2}$$

With this, the top hat will be centered at (x_{0b}, y_{0b}) with a bounding circle

$$(x_b - x_{0b})^2 + (y_b - y_{0b})^2 = a_0^2. \tag{5. 3}$$

The ellipse is given by

$$\frac{x_b^2}{A^2} + \frac{y_b^2}{B^2} = 1 \tag{5. 4}$$

or, in a form which will be useful shortly, in terms of the parameter ϕ as

$$x_b(\phi) = A \cos \phi$$

$$y_b(\phi) = B \sin \phi \ . \tag{5. 5}$$

The goal is to find the simultaneous solutions to (5. 3) and (5. 4) when they number two. In the cases of zero or one solutions, the line integral evaluates to zero. The cases of three, four, or infinite numbers of solutions will not occur because the antennas are not allowed to enter the ground patch. A solution method is to solve (5. 4) for y_b, expand (5. 3), and substitute, yielding

$$\underbrace{\left(1 - \frac{B^2}{A^2}\right)}_{A_1} x_b^2 \underbrace{- 2x_0}_{B_1} x_b \underbrace{+ x_0^2 + B^2 + y_0^2 - A_0^2}_{C_1} = 2y_0 B \sqrt{1 - \frac{x_b^2}{A^2}} \ .$$

Squaring both sides and dividing by A_1 results in a quartic in x_b,

$$x_b^4 + \underbrace{\frac{2B_1}{A_1}}_{a_1} x_b^3 + \underbrace{\left(\frac{2C_1}{A_1} + \frac{B_1^2}{A_1^2} + \frac{4y_{0b}^2 B^2}{A^2 A_1^2}\right)}_{b_1} x_b^2 + \underbrace{\frac{2B_1 C_1}{A_1^2}}_{c_1} x_b + \underbrace{\frac{C_1^2 - 4y_{0b}^2 B^2}{A_1^2}}_{d_1} = 0 , \tag{5. 6}$$

which can be solved by standard algebraic or numerical methods. (The algebraic method is used here, although, since many solutions are to be found, each of which differs from the last by a small amount, it might be more efficient to use numerical methods.) Each solution for x_b must be matched with the appropriate value of y_b because the y_b-value can be either positive or negative and still satisfy (5. 4). The result of doing this is two points, (x_{1b}, y_{1b}) and (x_{2b}, y_{2b}), which correspond to the desired angles θ_1 and θ_2.

Although it is not used explicitly in this dissertation, it may on occasion be necessary to have the above result specialized to the case of monostatic SAR, i.e ., when $A = B = R$. A separate derivation in which (5. 4) is so modified yields the quadratic

$$x^2 + b_2 x + c_2 = 0$$

in which

$$b_2 = \frac{2B_2C_2}{B_2^2 + 4y_0^2}$$

$$c_2 = \frac{C_2^2 - 4y_0^2R^2}{B_2^2 + 4y_0^2}$$

and

$$B_2 = -2x_0$$

$$C_2 = x_0^2 + R^2 + y_0^2 - a_0^2.$$

The second step is to find the indicated arc length in Fig. 5. 1. From elementary calculus, the formula for a line integral along a curve C of a function f, where ds is an increment of distance along the curve and when the curve can be expressed in terms of a parameter ϕ such as in (5. 5), is

$$\int_C f(x_b, y_b)\, ds = \int_{\phi_1}^{\phi_2} f\left[x_b(\phi), y_b(\phi)\right] \left[\left(\frac{dx_b}{d\phi}\right)^2 + \left(\frac{dy_b}{d\phi}\right)^2\right]^{1/2} d\phi \tag{5. 7}$$

where $\phi_1 \le \phi \le \phi_2$. For the present problem involving a unit top hat, if ϕ_1 and ϕ_2 can be made to correspond to θ_1 and θ_2, then $f(x_b, y_b) = 1$. Taking the indicated derivatives, the above becomes

$$\int_{\phi_1}^{\phi_2} \left(A^2 \sin^2\phi + B^2 \cos^2\phi\right)^{1/2} d\phi$$

$$= \int_{\phi_1}^{\phi_2} \left[A^2 \sin^2\phi + B^2 \left(1 - \sin^2\phi\right)\right]^{1/2} d\phi$$

$$= B\int_{\phi_1}^{\phi_2} \left[1 - k^2 \sin^2\phi\right]^{1/2} d\phi$$

where

$$k^2 = \frac{B^2 - A^2}{B^2}$$

is the square of the eccentricity of the ellipse. The Legendre elliptic integral of the second kind is given by

$$E(\alpha, k) = \int_0^{\alpha} \left[1 - k^2 \sin^2\phi\right]^{1/2} d\phi$$

and so the "unattenuated" elliptic-arc projections of a top hat are given by

$$f_{eu}(B, X_t, Y_t, X_r, Y_r) = \begin{cases} B\left[E(\phi_2, k) - E(\phi_1, k)\right] & \text{if ellipse intersects circle} \\ 0 & \text{otherwise} \end{cases} \tag{5. 8}$$

where ϕ_1 and ϕ_2 are found by substituting the solutions from (5. 6) into (5. 5) so that

$$\tan\phi_1 = \frac{A}{B}\tan\theta_1$$

$$\tan\phi_2 = \frac{A}{B}\tan\theta_2 .$$

The notation $f_{eu}(B, X_t, Y_t, X_r, Y_r)$, although cumbersome, is intended to indicate the more complex functional dependence of the projections with bistatic SAR. The quantity B is roughly analogous to p in the straight-line and circular-arc projections.

This, in principle, solves the elliptical-arc projections of the test function when propagation attenuation is negligible. However, the problem remains of evaluating the elliptic integrals. An expedient way of doing this is to use software such as the subroutine el2 which is available in [88] and which is claimed to be state-of-the-art. This subroutine is more general than is needed for the current problem and has four arguments. The elliptical-arc projections can be expressed as

$$f_{eu}(B, X_t, Y_t, X_r, Y_r) = \begin{cases} B\left[\text{el2}\left(\tan\phi_2, k_c, 1, k_c^2\right) - \text{el2}\left(\tan\phi_1, k_c, 1, k_c^2\right)\right] & \text{if ellipse intersects circle} \\ 0 & \text{otherwise} \end{cases}$$

where

$$k_c^2 = 1 - k^2 ,$$

as long as both θ_1 and θ_2 fall in the first quadrant. Some extension of the canned software is required here, but it is eased by the use of the complete elliptic integral of the second kind,

$$E(k) = E\left(\frac{\pi}{2}, k\right).$$

All details of this modification are incorporated into a program in Appendix D.

5.1.2 With propagation attenuation

Under far-field conditions, the attenuation suffered by a wave traveling from the transmitting antenna to a point (x_b, y_b) and returning to the receiving antenna, disregarding the reflectivity at that point, can be found from Fig. 5. 1 to be

$$\frac{1}{\left[x_b^2 + (F + y_b)^2\right]^{1/2}} \frac{1}{\left[x_b^2 + (F - y_b)^2\right]^{1/2}} \tag{5. 9}$$

which specializes to the monostatic case of $1/R^2$ when $F = 0$. This function is plotted over the ground patch in Fig. 5. 2 for the case of the transmitter at $(72, -50)$ and the receiver at $(72, 50)$ which are typical locations for the simulations which follow. In Fig. 5. 3 is shown the effect of this attenuation on the test function. As is obvious from the contours of Fig. 5. 2 (b), the attenuation is not constant along a single ellipse. In monostatic SAR, the attenuation is constant along circles of constant radius. It is because of this difference that the correction is more difficult for bistatic SAR.

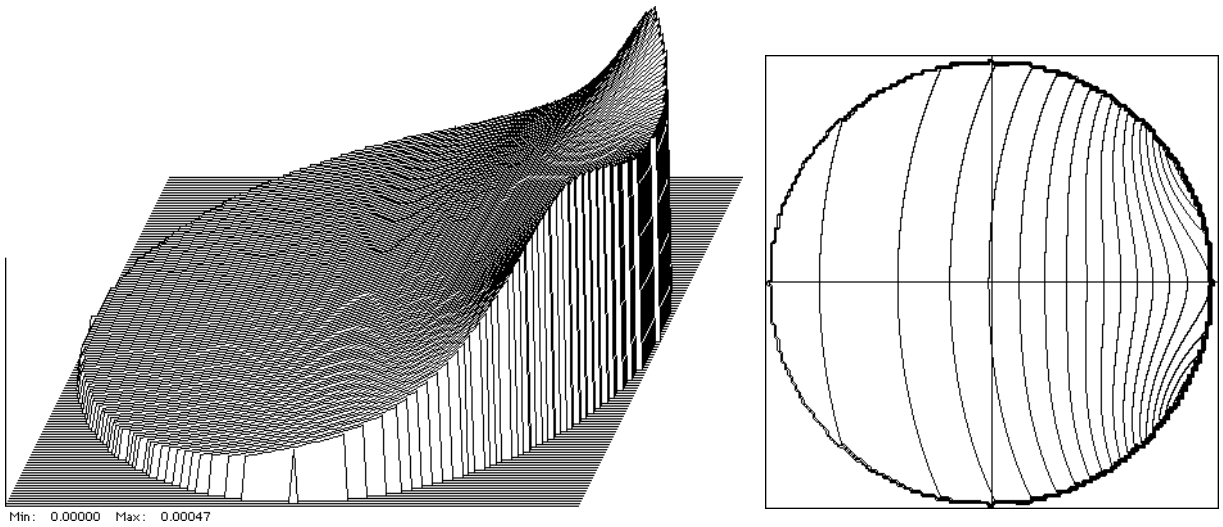
It may be useful to clarify the difficulty of the situation. In particular, one may wonder, since Fig. 5. 3 is made by multiplying the original test function by (5. 9) (as shown in Fig. 5. 2), a known quantity, why the inverse of (5. 9) could not be applied. The answer is twofold. First, the attenuation function (5. 9) rotates with the center of the bistatic system, effectively causing the ground patch to be affected by a different attenuation for each pulse—there is not a single function or inverse to deal with. Second, the modifed ground patch, e.g ., Fig. 5. 3, is not available for processing. Only the projections of the modified ground patch are available.

The derivation of the elliptical-arc projections of a unit top hat under the influence of the above attenuation is similar to the earlier unattenuated case and Fig. 5. 1 will again be useful. The angles θ_1 and θ_2 are found as before. With C restricted to the top hat by θ_1 and θ_2, the function $f(x_b, y_b)$ in (5. 7) is just the attenuation,

$$f(x_b, y_b) = \frac{1}{\left[x_b^2 + (F + y_b)^2\right]^{1/2}} \frac{1}{\left[x_b^2 + (F - y_b)^2\right]^{1/2}}.$$

Expanding each term of the denominator and substituting

$$F y_b = \sqrt{B^2 - A^2}\, B \sqrt{1 - \frac{x_b^2}{A^2}},$$



(a) (b)

Fig. 5. 2. Propagation attenuation plotted over the ground patch for the typical case of the transmitter at (72, –50) and the receiver at (72, 50). (a) Perspective plot. (b) Contours of constant attenuation.

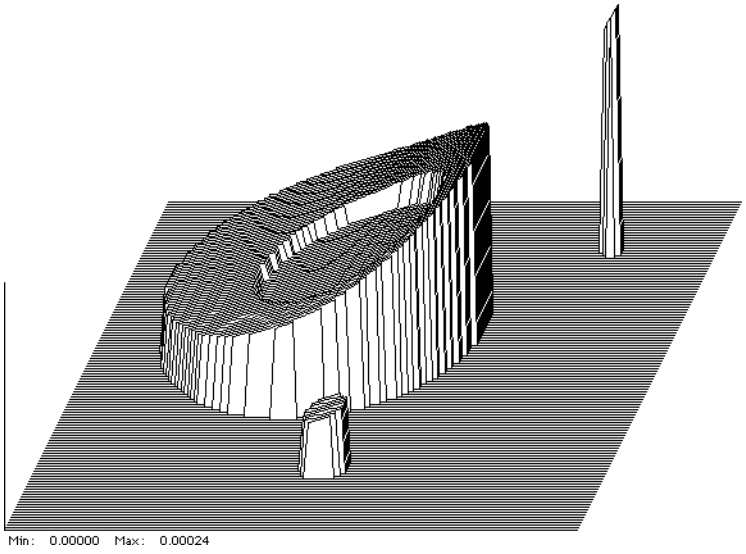


Fig. 5. 3. The effect of propagation attenuation on the test function: the function of Fig. 3. 1 multiplied by the function of Fig. 5. 2.

this can be written as

$$f(x_b, y_b) = \frac{1}{\left[x_b^2 + B^2 - A^2 + 2B\sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)} + B^2 - \frac{B^2}{A^2}x_b^2\right]^{1/2}} \cdot$$

$$\frac{1}{\left[x_b^2 + B^2 - A^2 - 2B\sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)} + B^2 - \frac{B^2}{A^2}x_b^2\right]^{1/2}}.$$

Collecting some terms, this becomes

$$f(x_b, y_b) = \frac{1}{\left[\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right) + 2B\sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)} + B^2\right]^{1/2}} \cdot$$

$$\frac{1}{\left[\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right) - 2B\sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)} + B^2\right]^{1/2}}.$$

Simplifying,

$$f(x_b, y_b) = \frac{1}{\left[B + \sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)}\right]\left[B - \sqrt{\left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)}\right]}$$

or

$$f(x_b, y_b) = \frac{1}{B^2 - \left(B^2 - A^2\right)\left(1 - \frac{x_b^2}{A^2}\right)}$$

$$= \frac{1}{A^2 + \left(\frac{B^2 - A^2}{A^2}\right)x_b^2}.$$

Recalling (5. 5) and further simplifying,

$$f(x_b, y_b) = \frac{1}{B^2\left[1 - \left(\frac{B^2 - A^2}{B^2}\right)\sin^2\phi\right]}.$$

The square root in (5. 7) evaluates to $B\left[1-k^2\sin^2\phi\right]^{1/2}$ as before, so the elliptical-arc projections which account for propagation attenuation are

$$f_{ea}(B, X_t, Y_t, X_r, Y_r) = \frac{1}{B}\int_{\phi_1}^{\phi_2} \frac{d\phi}{\left[1-k^2\sin^2\phi\right]^{1/2}}.$$

The Legendre elliptic integral of the first kind is

$$F(\alpha, k) = \int_0^{\alpha} \frac{d\phi}{\left[1-k^2\sin^2\phi\right]^{1/2}}$$

so the "attenuated" elliptical-arc projections are

$$f_{ea}(B, X_t, Y_t, X_r, Y_r) = \begin{cases} \frac{1}{B}\left[F(\phi_2, k) - F(\phi_1, k)\right] & \text{if ellipse intersects circle} \\ 0 & \text{otherwise} \end{cases}. \tag{5. 10}$$

In terms of the software subroutine el2, this is

$$f_{ea}(B, X_t, Y_t, X_r, Y_r) = \begin{cases} \frac{1}{B}\left[\text{el2}(\tan\phi_2, k_c, 1, 1) - \text{el2}(\tan\phi_1, k_c, 1, 1)\right] & \text{if ellipse intersects circle} \\ 0 & \text{otherwise} \end{cases}.$$

Specification of the various parameters is the same as for the case of unattenuated projections.

5.2 Algorithm Development and Simulations

This section will begin with a brief discussion of sampling issues that are involved in bistatic SAR, and continue with subsections devoted to combined discussions of new algorithms and related simulations. The discussion on sampling seems to be relevant to actual system design; it is not known what is usual practice. If the plane wave assumption is adequate, then sampling is simplified. However, it is necessary to resolve the sampling problem in some manner in order to carry out the simulations herein.

5.2.1 Sampling issues

Sampling in bistatic SAR means collecting a single projection from the continuous equations (5. 8) or (5. 10). The main problem is that the relationship between A and B, (5. 1), is nonlinear. With monostatic SAR, even with wavefront curvature, the sampling variable is p and there is generally no question as to how to arrange the sampling. As stated earlier, the sum of the distances from any point on the ellipse to the foci is $2B$. It therefore seems reasonable to sample uniformly in B, and that is the approach used here, since that is what would happen with hardware running at a constant clock rate. This leaves nonuniform sampling in A. The question remains as to what effect this has on reconstruction algorithms. This important matter will be discussed in the remainder of this chapter as it arises.

The next question is when to start and stop the sampling process. Referring to Fig. 5. 1, there is one member of the family of ellipses having the transmitting and receiving antennas as foci that is tangent to the ground patch at the near side, and another, larger, such ellipse which is tangent to the ground patch at the far side. Let the values of B for these two special ellipses be B_{min} and B_{max}, respectively. Assuming for the moment that these two quantities are known, given that sampling is uniform in B, and given that in general B_{min} and B_{max} will vary from pulse to pulse, the question then arises as to whether the sampling rate should remain constant from pulse to pulse, whether the same number of samples should be taken from each transmitted pulse, or some other scheme. The first method is appealing because of the simplicity in the sampling hardware. The second method is appealing because of the simplicity of the digital storage of the samples. The first method would require, at least conceptually, a two-dimensional storage array, each row of which would be filled in as each range profile (set of samples from a particular pulse) arrives. The array would have to be wide enough to accommodate the range profile having the largest number of samples. All other range profiles would not completely fill their respective rows, wasting memory and/or complicating addressing. The second method would require adjusting the sampling rate for each range profile. While there are no doubt other sampling schemes, the second method, that is, collecting the same number of samples for each range profile, is used in the simulations here because the storage problem for the other method is harder to implement than the sampling rate adjustment. There is one other possible problem with either of these sampling methods: increments in A can be quite large under some conditions, raising the possibility of undersampling portions of the ground patch with projections. This will be most likely to happen when the bistatic radar system is facing the ground patch "broadside," as opposed to "end on." This condition was not systematically monitored in the simulations, and it is not known whether it was responsible for any reconstruction artifacts. This is certainly an area for further investigation.

The problem of finding B_{min} and B_{max} could be tackled in a number of different ways. Since the radar presumably knows the trajectories of its own antennas, one approach might be to find explicit solutions by rearranging (5. 4) and the boundary of the ground patch expressed in bistatic

coordinates,

$$(x_b - X_b)^2 + (y_b - Y_b)^2 = L^2. \tag{5. 11}$$

This seems like an unnecessarily difficult problem in algebra. The approach used here takes advantage of the fact that the subroutine that is used to solve the quartic equation for the intersection of an ellipse with a circle returns the number of real-valued roots that were found. Instead of presenting it with the parameters of a top hat in order to let it help find θ_1 and θ_2, the circle representing the ground patch (5. 11) is substituted. This process is embedded in a numerical root-finder which adjusts the value of B until the equation

$$\text{Number of Roots} - 1 = 0$$

is satisfied, i.e., it searches for the two values of B, B_{min} and B_{max}, which satisfy the tangency condition. The root-finder uses a bisection method [88] which is said to be efficient in finding well-defined roots such as is the case here, since the left-hand side of the above equation can take only the whole-number values zero through three in the radar setting. Also, since the solutions for B_{min} and B_{max} normally change only slightly from look to look, the previous values can be used to begin the search for the current values. This is probably more efficient than any algebraic method.

5.2.2 Unattenuated propagation, simplified trajectories

The simplified trajectories used in this section are contrived but nevertheless useful for algorithm development. There are three different trajectories which differ by the initial positions of the transmitter and receiver. In all three cases, the transmitter and receiver travel concentric circles centered on the ground patch, and the angular rate of travel is the same for both. They both arrive back at their starting positions at the same time. These trajectories might be thought of as a "lock-step" type. Names for the three trajectories are given by their initial positions as follows:

Horizontal
- Transmitter at (172, 0)
- Receiver at (72, 0)

Vertical
- Transmitter at (72, -50)
- Receiver at (72, 50)

Oblique
- Transmitter at (144.61955, 44.21463)
- Receiver at (68.85394, -21.05076).

The value of F for all three is 50 and the receiver is positioned such that a line drawn through the receiver and parallel to the y-axis intersects the x-axis at 72.

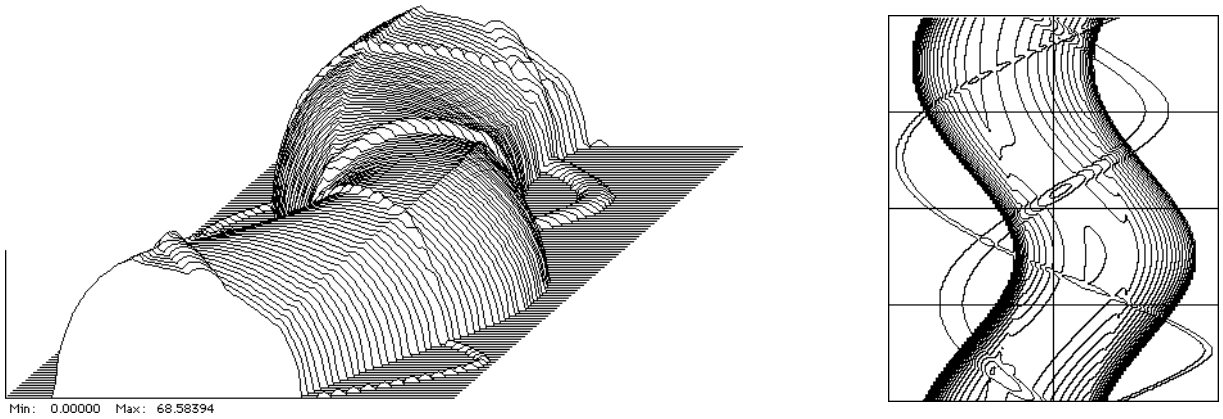
The projections for these three cases will be displayed first, since there will be a number of reconstructions to follow. These are, in the above order, Fig. 5. 4, Fig. 5. 5, and Fig. 5. 6. Although these plots appear similar, small systematic differences among them become important during reconstruction.

An example of reconstructing elliptical-arc data using unmodified (straight-line) convolution-backprojection is shown in Fig. 5. 7 , which is made from the data of Fig. 5. 5, the vertical case. A reconstruction from the horizontal data showed less defocusing on the central portions but more at the outer portions, although subjectively the overall distortion for that case was less.

The next two reconstructions, Fig. 5. 8 and Fig. 5. 9, are from the horizontal and vertical cases, respectively. The algorithm used is backprojection along elliptical arcs (the backprojection preceded, as always, by the standard convolution). The elliptical arcs are the same arcs from which the data were collected, only in the reconstruction plane instead of the ground patch plane. The correct variable in which to perform the one-dimensional interpolation is B, the uniform-sampling quantity, in all cases, here and later. One may feel inclined to interpolate in A for the vertical case, since it is A which seems to be "propagating" across the ground patch; however an interpolation in A, between projection samples, while yielding only slightly different interpolated values under most conditions, is not correct, as it is inconsistent with the physical nature of the problem.

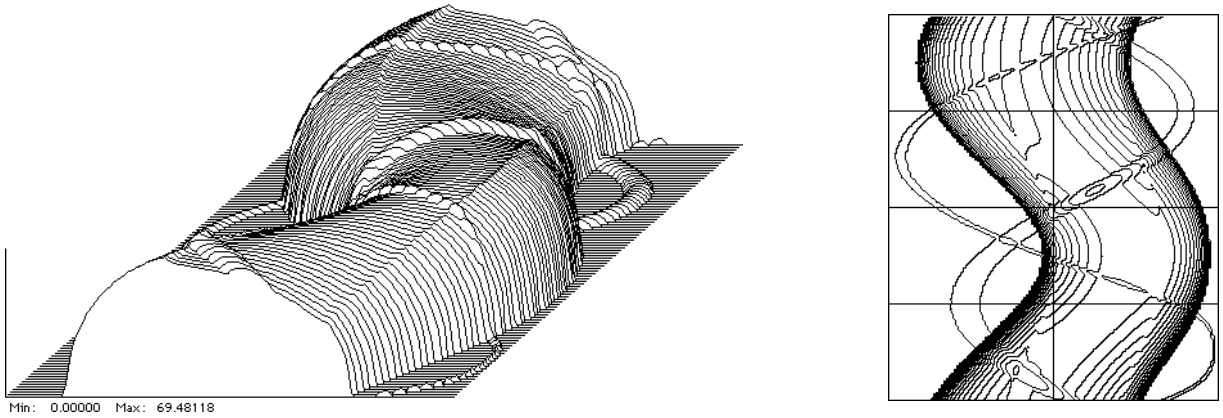
The quality of the two reconstructions, Fig. 5. 8 and Fig. 5. 9, is quite different. The horizontal case, Fig. 5. 8, is very good, apparently needing little improvement. However, Fig. 5. 9, the vertical case, shows some fairly serious gray-level artifacts, especially in the vicinity of the large top hat feature. A large trough has formed behind it, and there is an overshoot, a lip, on the upper portion of that top hat. Oddly, these two aberrations are next to one another, so clearly it is not possible to fix the problem by simply multiplying the image by some corrective function, at least not a function which is data-independent. Other low-level artifacts are visible as well.

Some insight into the above problem can be had by referring to Fig. 5. 10, which shows contours at equal increments of B (they are ellipses) with F fixed. The two circles represent the ground patches for the vertical and horizontal cases, both shown to scale and on the same plot for simplicity. Two differences are immediately obvious. For the horizontal case, the contours appear to have about the same distance between any two adjacent ones, and the spacing between them is roughly uniform across the patch; indeed, the spacing measured along the y_b-axis is exactly uniform, since this is the definition of B. This would appear to approximate the circular-arc case for these reasons. However, neither of these conditions holds as well for the horizontal case. The distance between contours, measured along the x_b-axis (A), increases much more rapidly for small magnitudes of x than for larger magnitudes. (In fact, near the line segment connecting the foci, the incremental increase in A with respect to B increases without limit as B approaches F.) In addition,



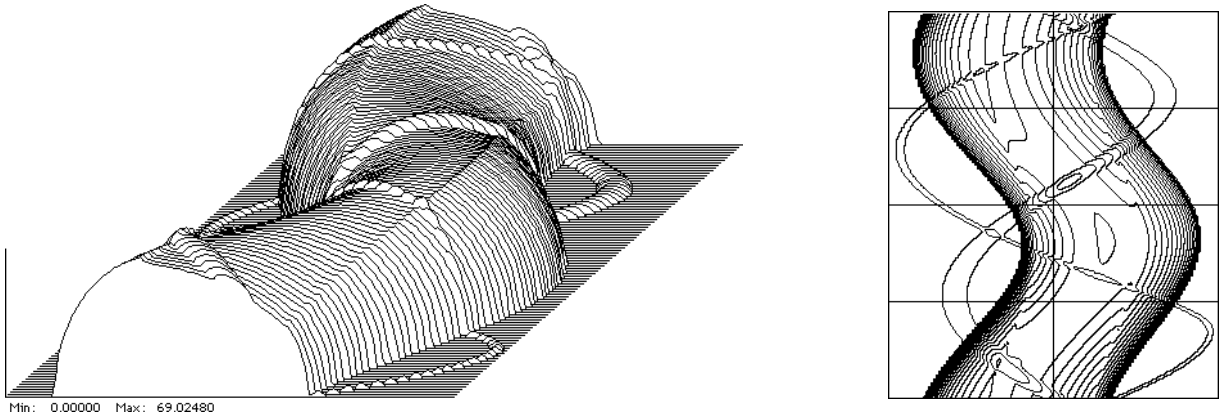
(a) (b)

Fig. 5. 4. Unattenuated elliptical-arc projections for a horizontal initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



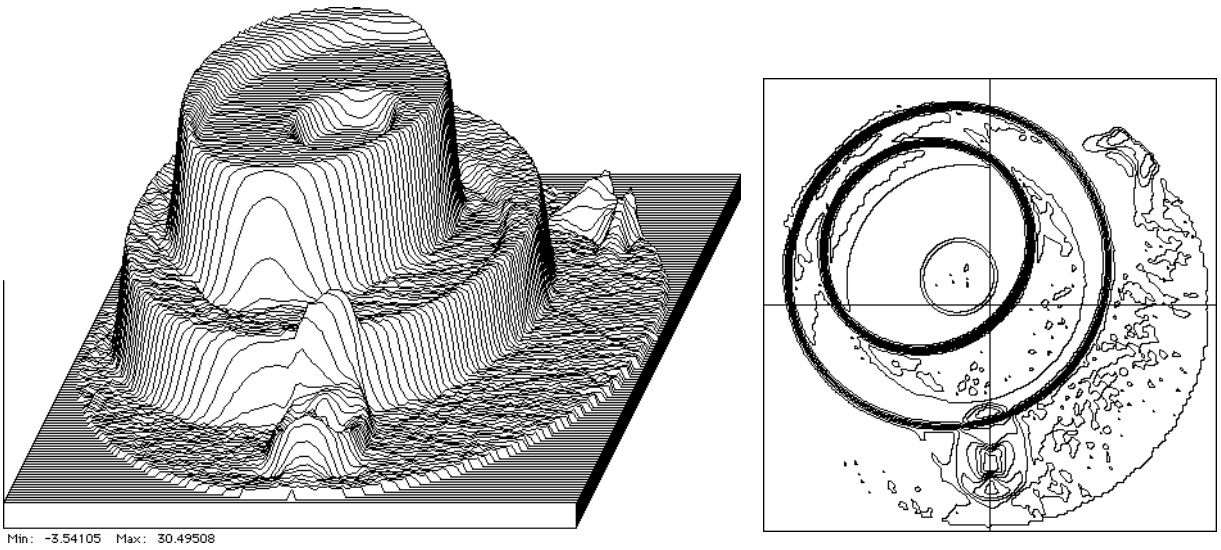
(a) (b)

Fig. 5. 5. Unattenuated elliptical-arc projections for a vertical initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 6. Unattenuated elliptical-arc projections for an oblique initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



(a) (b)

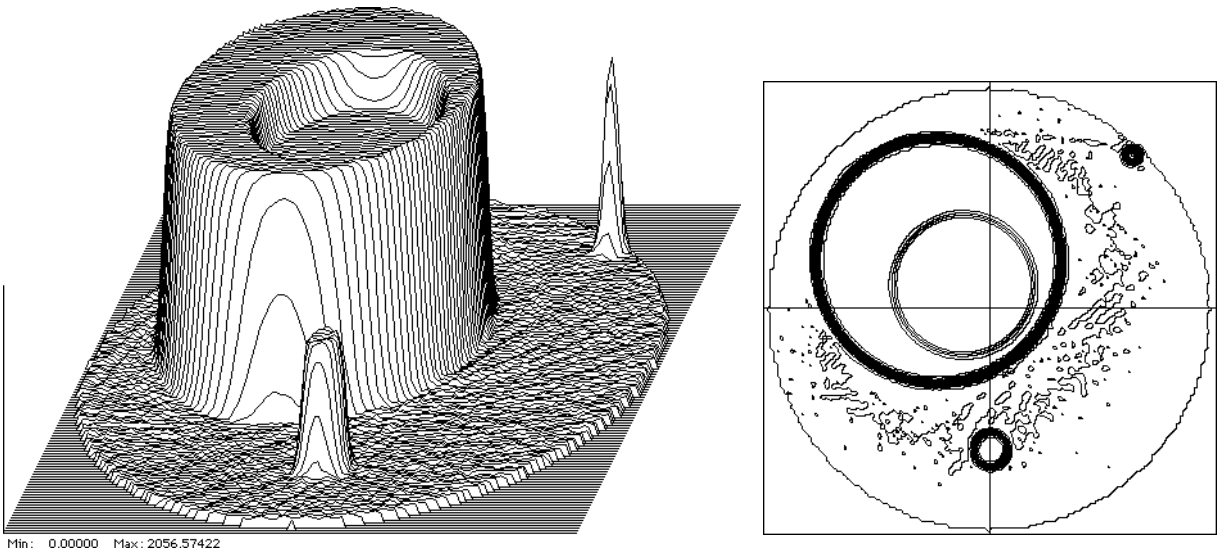
Fig. 5. 7. Reconstruction from elliptical-arc projections, vertical case, using unmodified convolution-backprojection. (a) Perspective plot. (b) Contour plot.

the variation in local curvature of the elliptical contours across the patch appears greater. Finally, one notices that the elliptical arcs which are nearest the bistatic origin and which intersect the ground patch are longer for the vertical case than the horizontal case because they have less curvature.

In light of the preceding discussion, the problem is to make an adjustment to the algorithm which has only a small effect on the reconstruction of the horizontal case while improving the reconstruction of the vertical case. Note that each point in the (x_b, y_b) plane has a unique ellipse from the set of confocal ellipses which passes through it, and that B and A are determined uniquely. With this in mind, the derivative of A with respect to B over the (x_b, y_b) plane is found from (5. 1) to be

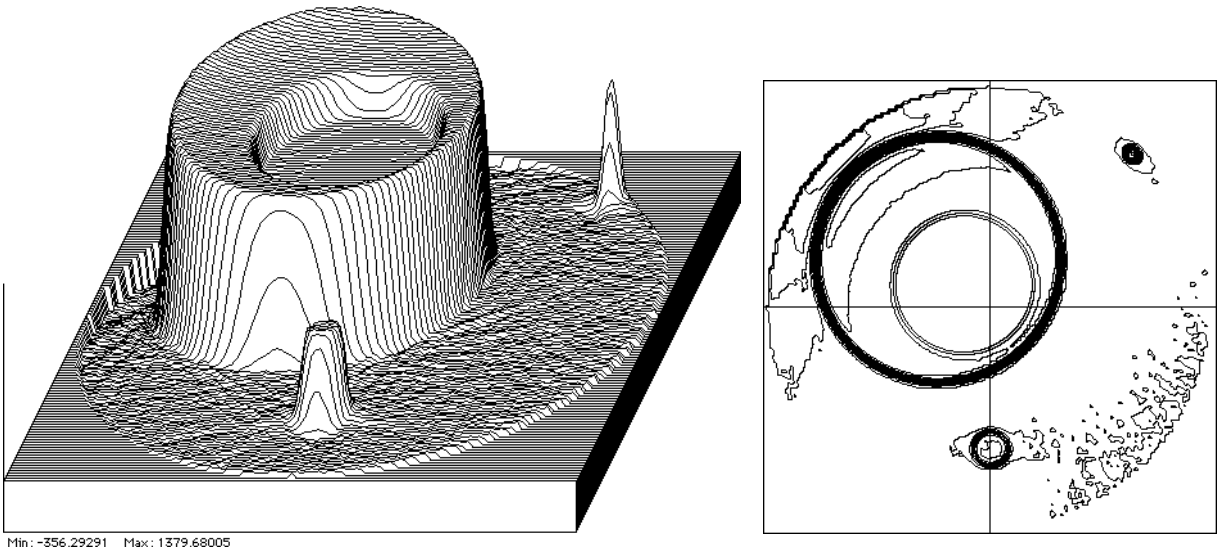
$$\frac{dA}{dB} = \frac{B}{A}. \tag{5. 12}$$

Now, the question as to how to apply (5. 12) arises. Clearly, it will have to be applied in some form to affect the backprojection process, resulting in weighted backprojections. Shown in Fig. 5. 11 are two elliptical-arc projections of only the largest top hat after filtering; Fig. 5. 11 (a) is the first filtered projection, the projection taken when the center of the bistatic system was at zero degrees, and Fig. 5. 11 (b) is the 74^{th} filtered projection, taken when the center of the bistatic system was nearest to 135 degrees, i.e ., at its point of closest approach to the large top hat. The plot depicted in Fig. 5. 11 (a) is typical of most of the filtered projections dealt with so far, including ones from straight-line and circular-arc projections, in that there is a relatively flat part which



(a) (b)

Fig. 5. 8. Reconstruction from elliptical-arc projections, horizontal case, using unweighted elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 9. Reconstruction from elliptical-arc projections, vertical case, using unweighted elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.

spans the top hat when backprojected, and sharp, negative-valued dips on either side. These dips serve to "chisel away" the parts of the other backprojected functions which make contributions to the image being reconstructed outside of the top hat boundary. The plot in Fig. 5. 11 (b) shows exaggerated overshoots on both sides of the negative-going transition, near index number 60. Comparing the projections of Fig. 5. 4 and Fig. 5. 5, one also notices the steeper sides of the projections for the vertical case in the vicinity of the 74 th one, caused by the flattened (and thus lengthened) nature of the elliptical arcs, as mentioned before. These two statements are consistent given the

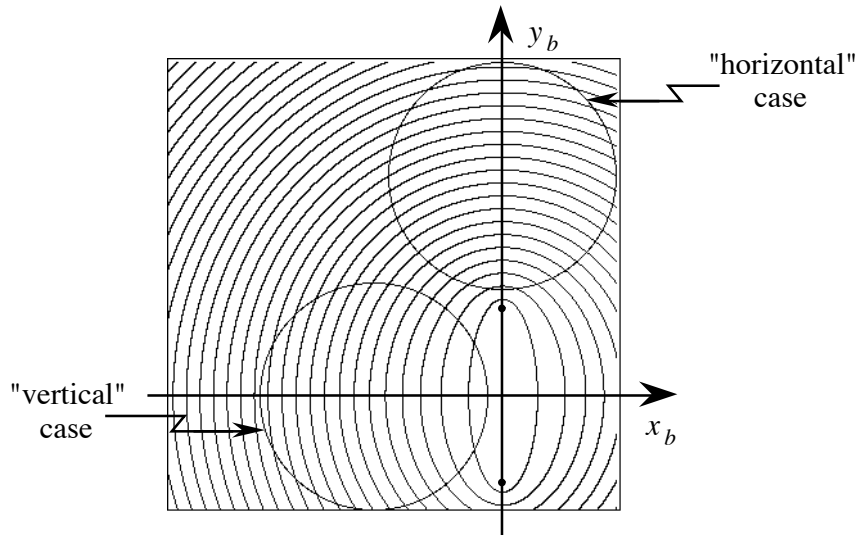


Fig. 5. 10. Circles are locations of the ground patches for the horizontal and vertical initial positions of the bistatic system. Ellipses are contours of equal B showing the different shapes and spacings within each ground patch.

impulse response of the filter. It appears, therefore, that whatever weighting is applied to the back-projection should tend to diminish the effect of these exaggerated overshoots and should tend to leave the others unaffected.

The inverse of (5. 12) is plotted over the ground patch in Fig. 5. 12 for the bistatic system in its initial vertical position and its horizontal initial position. If applied as a weighting during back-projection, both plots show a function which has the desired effect on the exaggerated overshoots of Fig. 5. 11 (b) while leaving the corresponding portion of Fig. 5. 11 (a) relatively unchanged. Moreover, the overall variation of Fig. 5. 12 (b) is substantially less than that of Fig. 5. 12 (a), indicating that a reconstruction using this function on the horizontal data would be relatively unaffected. A reconstruction of the vertical case using this weighting indeed repaired most of the problems with Fig. 5. 9. However, it was found that using A^2/B^2 was even more effective. Reconstructions using this weighting for the horizontal, vertical, and oblique cases are shown in Fig. 5. 13, Fig. 5. 14, and Fig. 5. 15 respectively. The horizontal and oblique cases are quite good. The vertical case has a small vestige of its former problems, but is much improved and nearly as good as the other two.

5.2.3 Other weighting methods

As discussed above, an inspection of Fig. 5. 10 suggests two other weighting methods, in addition to A^2/B^2 . The first of these methods is to weight according to the local curvature of the ellipse which passes through the coordinates of the pixel being reconstructed, since it appears that the increase in projection values for the vertical case compared to the horizontal case is due to the

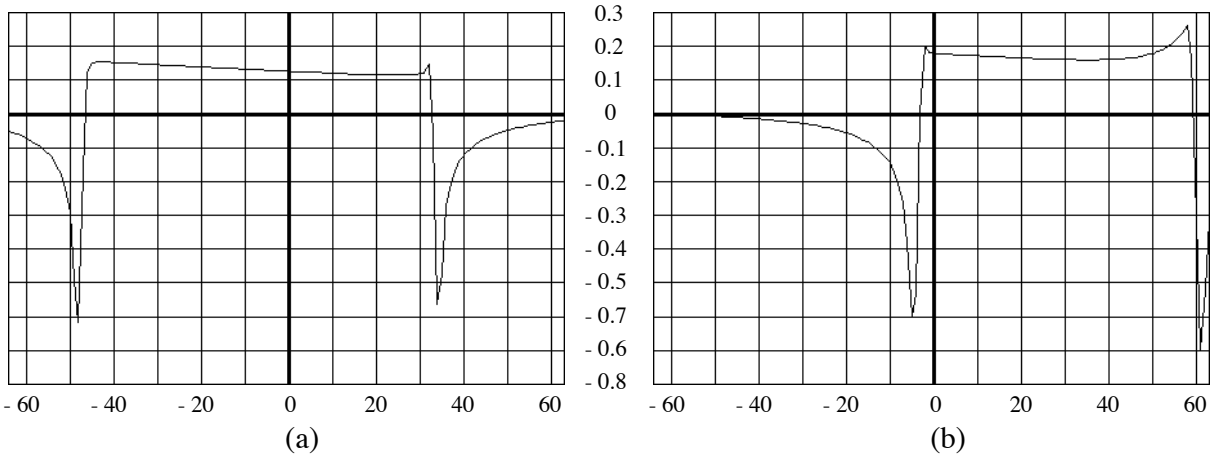


Fig. 5. 11. Filtered projections of only the large top hat, of the test function with the bistatic SAR in the vertical initial position. (a) For the bistatic system at zero degrees, projection number 1. (b) For the bistatic system at 135 degrees, projection number 74 out of 198.

flattening of some of the elliptical arcs in the former. Using the parametric form (5. 5) for the ellipse, this curvature can be found to be

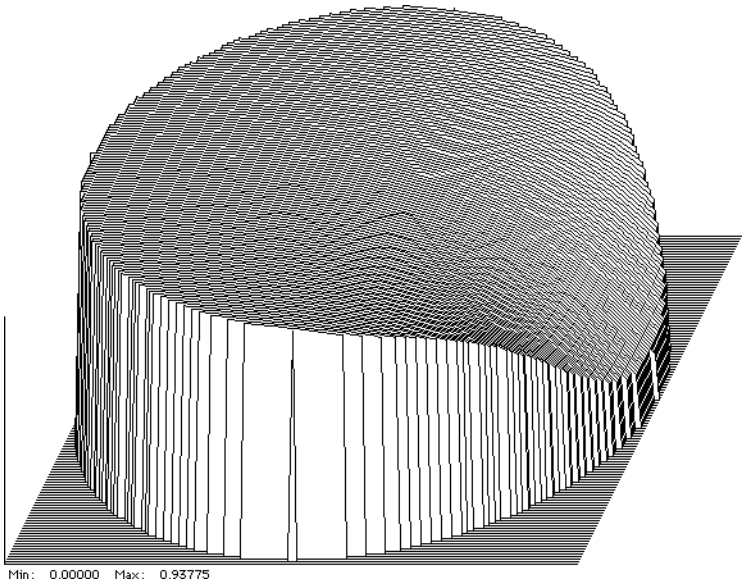
$$\kappa\left(x_b, y_b\right) = \frac{A^4 B^4}{\left(B^4 x_b^{\,2} + A^4 y_b^{\,2}\right)^{3/2}}.$$

Although this looks promising, several reconstructions done using the projections from the vertical, unattenuated case with simplified trajectories, as in the preceding section, were not as good as the ones of Fig. 5. 13, Fig. 5. 14, and Fig. 5. 15, although some improvement was noted.

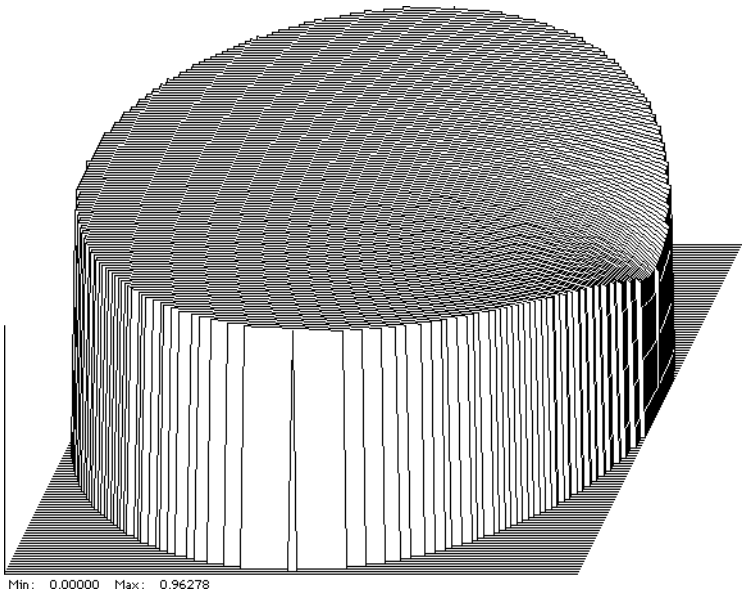
Another method of weighting which is a generalization of A^2/B^2 was also tried, with similar results (essentially identical in some cases). This weighting scheme requires the introduction of the confocal coordinate system, which will also be found to be useful in a later section dealing with reconstruction under conditions of attenuated propagation.

The confocal coordinate system is an orthogonal system as shown in Fig. 5. 16 which shows confocal ellipses as contours of constant v and confocal hyperbolas as contours of constant u, comprising the coordinate pair (u, v). (Small anomalies of symmetry in Fig. 5. 16 are plotting artifacts.) A relationship between rectangular coordinates (here, the bistatic coordinates) and confocal coordinates is

$$\frac{x_b^2}{F^2\left(v^2 - 1\right)} + \frac{y_b^2}{F^2 v^2} = 1$$



(a)



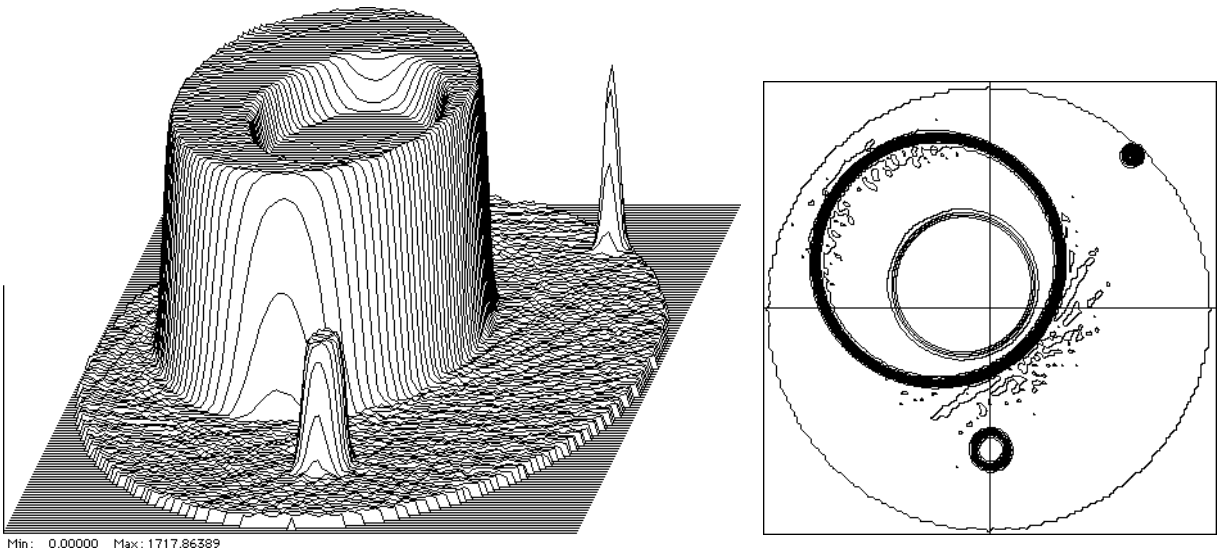
(b)

Fig. 5. 12. A plot of the confocal ellipse parameter A/B over the ground patch. This function is used to improve bistatic SAR reconstructions. (a) For vertical initial position. (b) For horizontal initial position.

$$\frac{y_b^2}{F^2\left(1-u^2\right)} - \frac{x_b^2}{F^2 u^2} = 1.$$

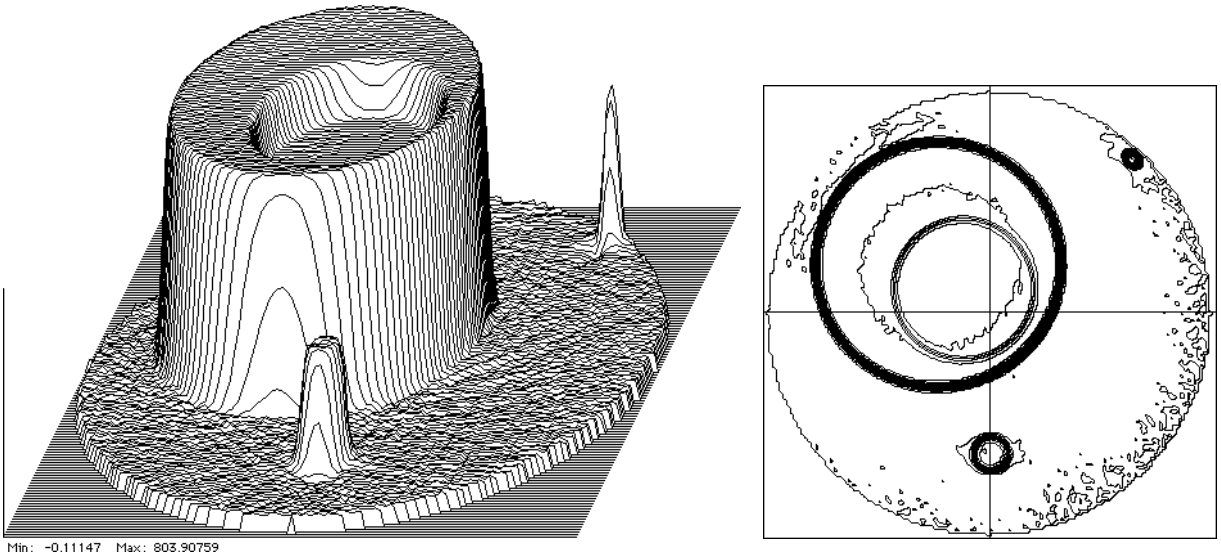
Also of use are the relationships

$$A^2 = F^2 \sinh^2 V = F^2\left(v^2 - 1\right) \quad B^2 = F^2 \cosh^2 V = F^2 v^2$$



(a) (b)

Fig. 5. 13. Reconstruction of the horizontal case using A^2/B^2 weighting during backprojection. (a) Perspective plot. (b) Contour plot.



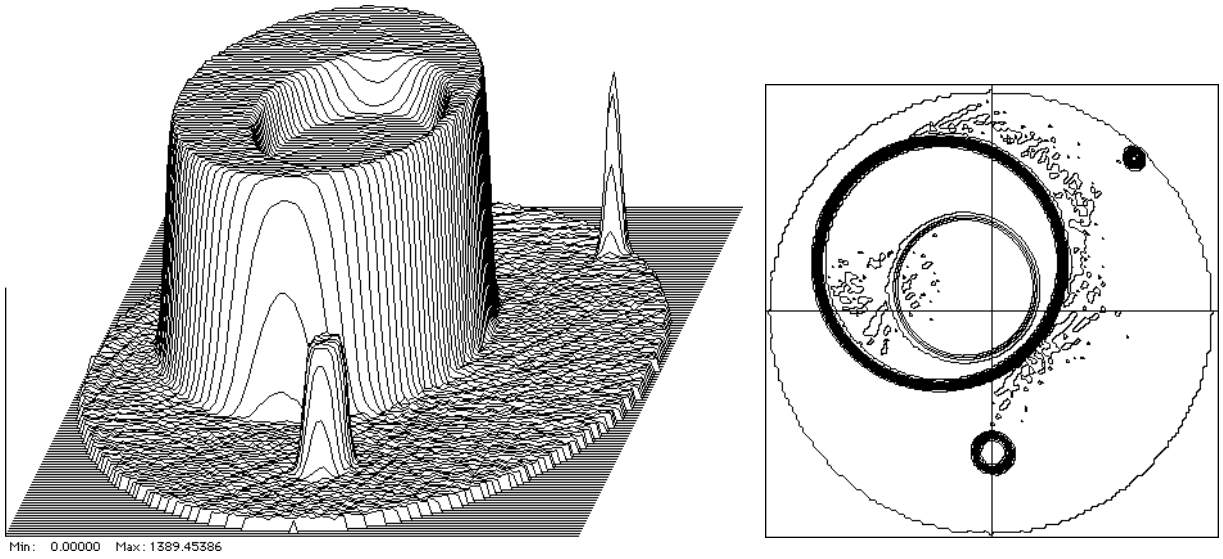
(a) (b)

Fig. 5. 14. Reconstruction of the vertical case using A^2/B^2 weighting during backprojection. (a) Perspective plot. (b) Contour plot.

$$C^2 = F^2 \sin^2 U = F^2\left(1 - u^2\right) \quad D^2 = F^2 \cos^2 U = F^2 u^2 \tag{5. 13}$$

which introduce the new hyperbolic parameters C and D which are analogous to the elliptic parameters A and B, and the intermediate variables U and V. The quantities C and D are related by

$$F^2 = C^2 + D^2$$



(a) (b)

Fig. 5. 15. Reconstruction of the oblique case using A^2/B^2 weighting during backprojection. (a) Perspective plot. (b) Contour plot.

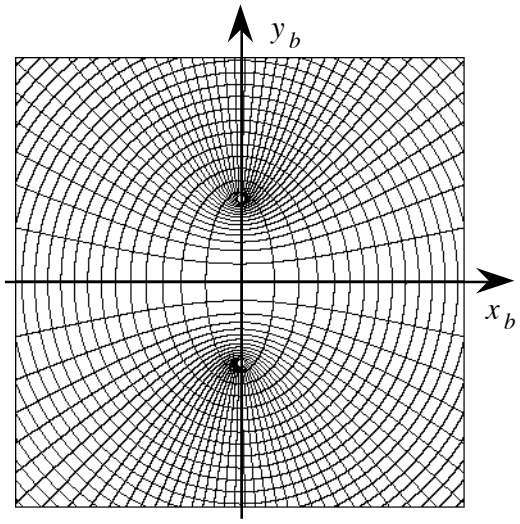


Fig. 5. 16. The confocal coordinate system superimposed on the bistatic coordinate system, (x_b, y_b). The two white dots are the locations of the foci.

and $2C$ is the difference of distances of any point on a hyperbola from the foci. The rectangular coordinates may conveniently be written as

$$x_b = F\cos U \ \sinh V$$

$$y_b = F\sin U \ \cosh V. \tag{5. 14}$$

An explicit solution for the inverse, u and v in terms of x_b and y_b, is messy and not needed here.

Returning to the problem of the alternate weighting scheme, let an incremental distance along a particular hyperbola at coordinates (u, v) be called ds. Then a quantity which expresses the non-linear expansion rate of the ellipses with respect to the uniformly-sampled quantity B is the derivative ds/dB. More formally, let $\mathbf{r}$ be the position vector defined by the functions (5. 14), $\mathbf{r} = \mathbf{r}(u, v)$, or, more conveniently, $\mathbf{r} = \mathbf{r}(C, B)$, with the aid of (5. 13). Then, with C held constant,

$$ds = \left|\frac{\partial \mathbf{r}}{\partial B}\right| dB$$

where

$$\left|\frac{\partial \mathbf{r}}{\partial B}\right| = \sqrt{\left(\frac{dx_b}{dB}\right)^2 + \left(\frac{dy_b}{dB}\right)^2} \ .$$

This quantity is most easily found using

$$\left|\frac{\partial \mathbf{r}}{\partial B}\right| = \left|\frac{\partial \mathbf{r}}{\partial V}\frac{dV}{dB}\right|$$

where

$$\left|\frac{\partial \mathbf{r}}{\partial V}\right| = \sqrt{\left(\frac{dx_b}{dV}\right)^2 + \left(\frac{dy_b}{dV}\right)^2} \ .$$

The calculation is straightforward and results in

$$\left|\frac{\partial \mathbf{r}}{\partial B}\right| = \frac{\sqrt{A^2C^2 + B^2D^2}}{AF}. \qquad (5.\ 15)$$

This is seen to be a generalization of the A^2/B^2 weighting. Along the x_b-axis, $u = 1$, $D^2 = F^2$, and $C^2 = 0$, so that

$$\frac{ds}{dB} = \frac{B}{A} = \frac{dA}{dB} \quad \text{(on the } x_b\text{-axis)},$$

which agrees with the earlier result (5. 12). Along the y_b-axis such that $|y_b| > F$, the conditions $u = 0$, $D = 0$, and $C^2 = F^2$ hold, so that

$$ds = dB \quad (\text{on the } y_b\text{-axis}, |y_b| > F)$$

as expected, since s and B are measuring the same thing.

Another topic related to these weighting styles is whether to apply the weighting before or after the filtering. Both methods were tried in numerous examples, and in all cases the best results

were obtained by applying the weighting after filtering. This is fortuitous because applying the weighting before filtering is much more computation-intensive, although there are some suboptimal ways of doing this that are more efficient. The style of algorithm which applies the weighting before filtering is helpful in the case of correcting for attenuated propagation, and will be described later. For present purposes, comparisons were made with low-resolution reconstructions in order to speed up the simulation process. The low-resolution plots were quite adequate for the job, however.

5.2.4 Unattenuated propagation, extended trajectories

The trajectories used so far, although useful in algorithm development, are rather contrived. This section will introduce a new family of trajectories and present a new problem which arises and its solution.

To define the trajectories, it is first necessary to set up some preliminary parameters. Towards this end, let (X_{t0}, Y_{t0}) and (X_{r0}, Y_{r0}) be the initial locations of the transmitter and receiver, respectively. Then the corresponding initial distance to the origin of the bistatic system and its initial tilt angle relative to the *x*-axis are

$$R_0 = \frac{1}{2}\sqrt{(X_{t0}+X_{r0})^2+(Y_{t0}+Y_{r0})^2} \quad \text{and} \quad \theta_{b0} = \tan^{-1}\left(\frac{X_{t0}-X_{r0}}{Y_{r0}-Y_{t0}}\right)$$

where Fig. 5. 1 applies. Two changes will be made to the simple trajectories used earlier which will give rise to many varied trajectories. First, the bistatic system will be allowed to turn on its origin at some rate ω, either positive or negative. Second, the focal distance F will be modulated with look angle. This can greatly reduce the regularity of the trajectories and relieve some of the contrivance of the earlier ones. Let the initial focal distance be

$$F_0 = \frac{1}{2}\sqrt{(X_{t0}-X_{r0})^2+(Y_{t0}-Y_{r0})^2}\,.$$

The trajectories of the transmitter and receiver in image coordinates, with n indexing the pulse transmission number, are then defined as

$$X_{tn} = R_0\cos\Theta_n + F_n\sin(\omega\Theta_n+\theta_{b0})$$

$$Y_{tn} = R_0\sin\Theta_n - F_n\cos(\omega\Theta_n+\theta_{b0})$$

$$X_{rn} = R_0\cos\Theta_n - F_n\sin(\omega\Theta_n+\theta_{b0})$$

$$Y_{rn} = R_0\sin\Theta_n + F_n\cos(\omega\Theta_n+\theta_{b0})$$

where

$$F_n = F_0\left[1 - M + M\cos\left(m\,\Theta_n\right)\right]$$

with Θ_n the angle from the ground patch origin to the bistatic origin, M a parameter relating the depth of the modulation of F, and m describing the frequency of the modulation of F, independent of ω. Normally, one would have $0 \le M \le 1/2$. Also, m would have to take on integer values in order for the end points of the trajectories to coincide with the initial points; this is certainly not a constraint of the simulations, however.

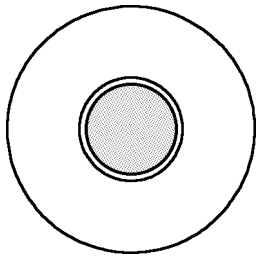
For simulation purposes, two new trajectories will be used and results compared to one of the earlier, simpler, trajectories. Since there are now other parameters to adjust in order to make different trajectories, the initial positions assume less relative importance. Accordingly, they will be taken to be the same as the horizontal case used earlier. The three trajectories, beginning with the simple one, are defined and named below; the diagrams are to scale with the shaded circle representing the ground patch.

Type 0

$\omega = 1$

$M = 0$

m = don't care

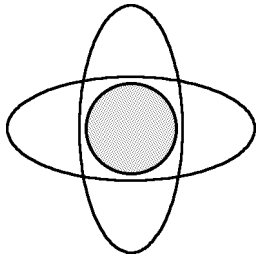


Type 1

$\omega = -1$

$M = 0$

m = don't care

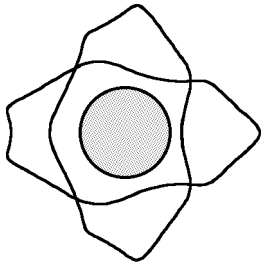


Type 2

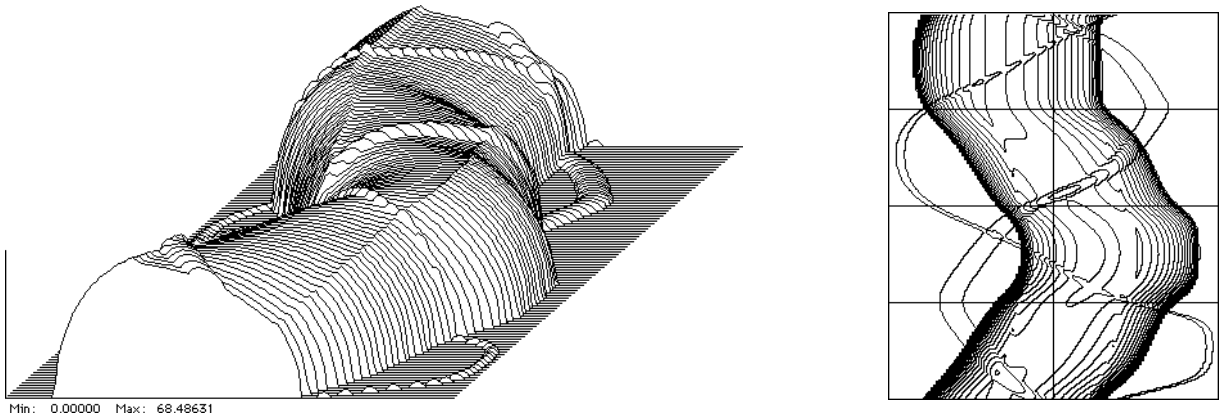
$\omega = -1$

$M = 0.25$

$m = 5$

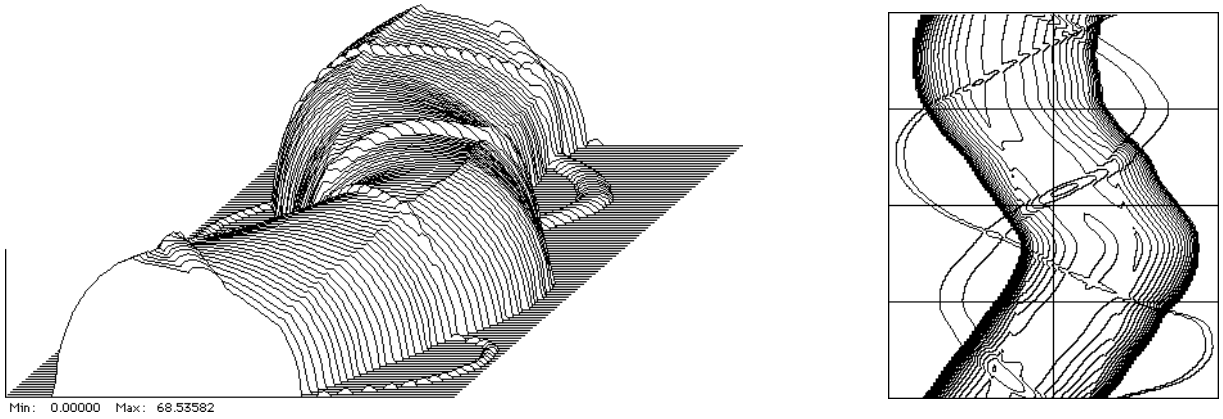


Since the Type 0 simulation has already been done, the simulations for Type 1 and Type 2 will be presented. Shown in Fig. 5. 17 and Fig. 5. 18 are the projections for these cases, respectively. A reconstruction for the Type 1 data which used the weightings of (5. 15) (in inverse square form) is shown in Fig. 5. 19. Apparently, the rotation ω has caused the alternating raised and depressed areas around the bases of the top hats, most obviously the large one and to a lesser



(a) (b)

Fig. 5. 17. Ellipitical-arc projections for the Type 1 trajectory. (a) Perspective plot. (b) Contour plot.

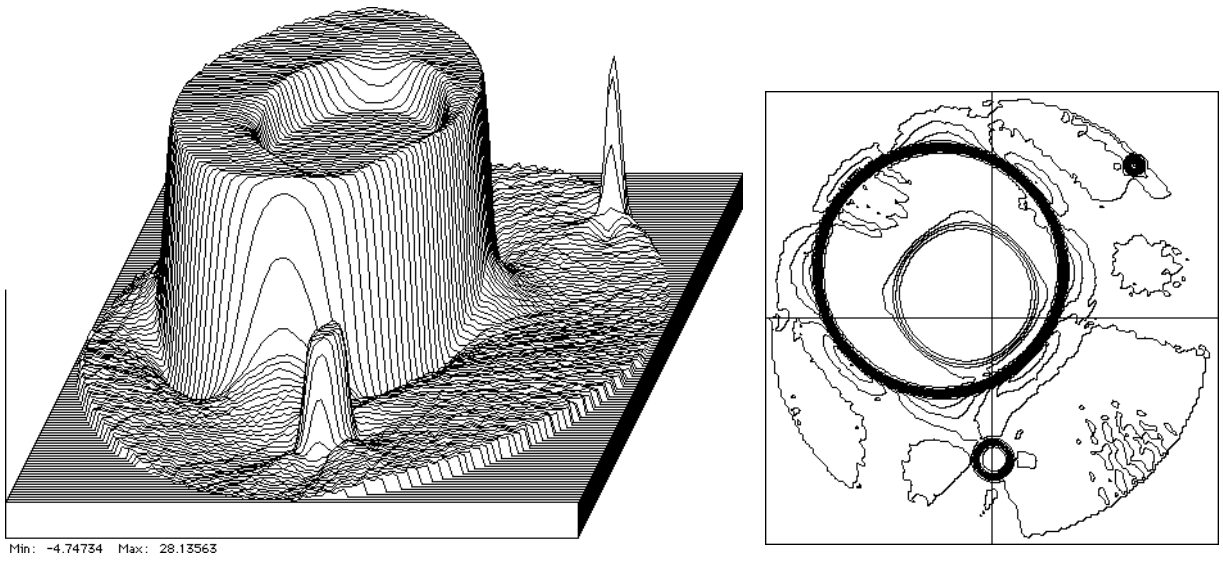


(a) (b)

Fig. 5. 18. Ellipitical-arc projections for the Type 2 trajectory. (a) Perspective plot. (b) Contour plot.

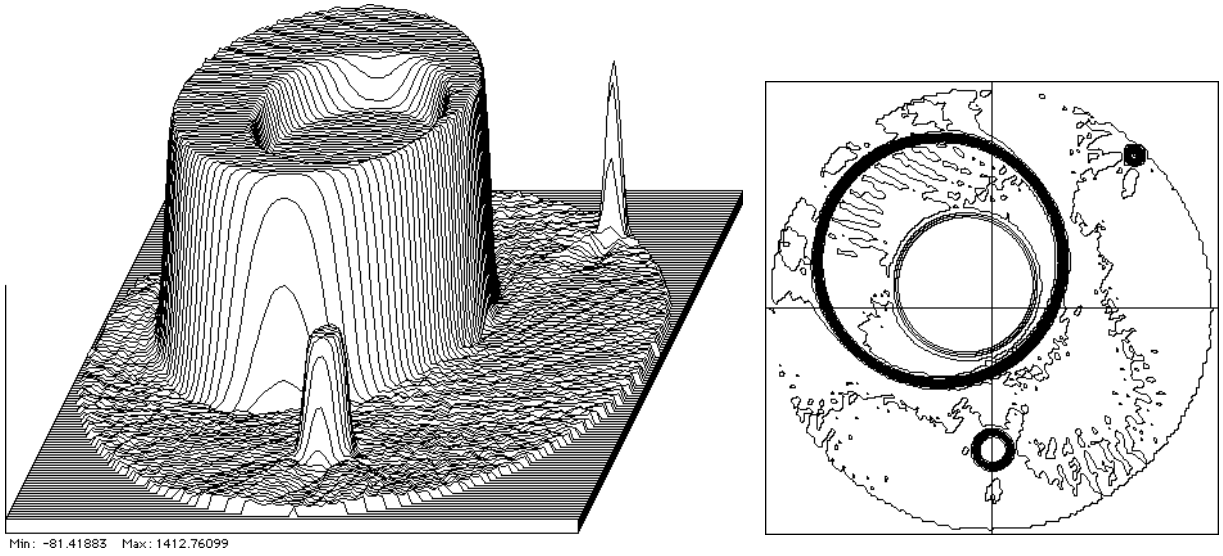
degree on the others. Referring again to the chiseling action of the filtered projections illustrated in Fig. 5. 11, it appears that a larger weight should be applied to the filtered projections from angles 0°, 90°, 180°, and 270° in order to reduce the height of those raised areas, while the filtered projections from angles 45°, 135°, 225°, and 315° should get less weight. One might think that perhaps a sinusoidal weight plus a constant (so as to not go negative, or even close to zero) might be a good weighting scheme, where the sinusoidal function goes through two full cycles during one circumnavigation by the bistatic system. As will be seen shortly, this is in fact true. However, it is desirable to develop a weighting scheme that is not so highly dependent upon the trajectory so that perhaps the same scheme will work for another type of trajectory, such as the Type 2 trajectory. Towards this end, it can be shown for the Type 1 trajectory that B_{min} and B_{max} both have the form of a raised, double-frequency sinusoid. It was found that an overall weight

$$B_{max} - \alpha B_{min} \tag{5. 16}$$



(a) (b)

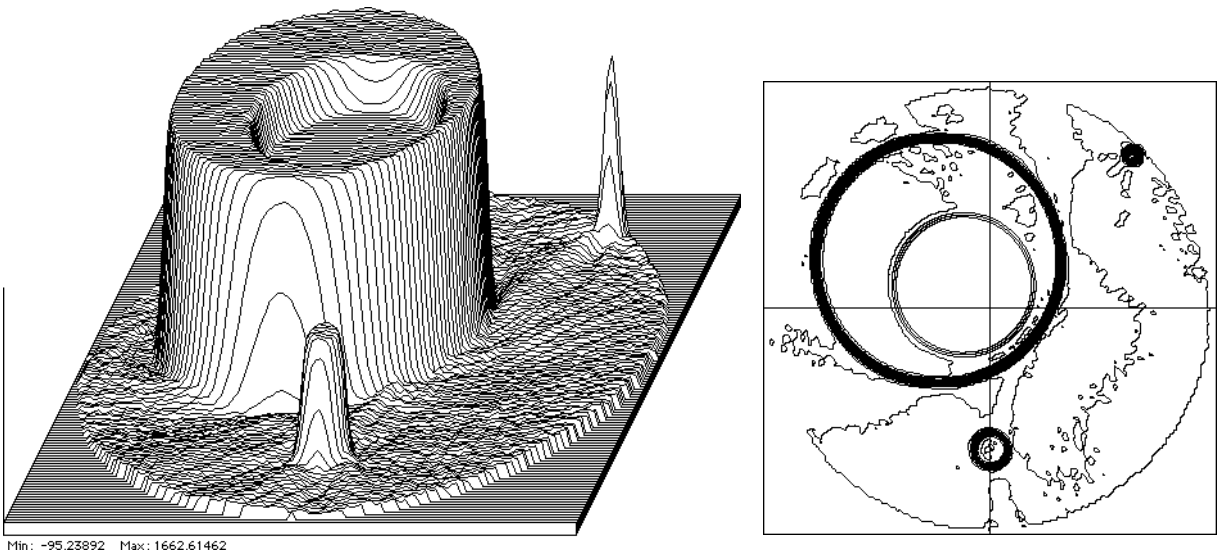
Fig. 5. 19. Reconstruction of Type 1 elliptical-arc projections without using a per-projection weighting. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 20. Reconstruction of Type 1 elliptical-arc projections using a per-projection weighting. (a) Perspective plot. (b) Contour plot.

with a value of 2 for α worked well, and a reconstruction based on this weight and the backprojection weighting of (5. 15) is shown in Fig. 5. 20. Values of α very near to 2 (1.95 and 2.05) were tried and a distinct deterioration in the reconstructions was noticed compared to the reconstruction made with $\alpha = 2$. The weighting (5. 15) was found to be slightly better for this series of reconstructions than the weighting of (5. 12).



(a) (b)

Fig. 5. 21. Reconstruction of Type 2 elliptical-arc projections using a per-projection weighting. (a) Perspective plot. (b) Contour plot.

It should be noted that the reconstruction for the Type 0 case remains unaffected by the per-pulse weighting of (5. 16) since B_{max} and B_{min} are constant.

A reconstruction of the Type 2 data of Fig. 5. 18 is shown in Fig. 5. 21. The algorithm is the same as that used to make Fig. 5. 20 in all respects.

With the more complicated trajectories for the radar, such as Type 1 and Type 2 used here, the issue of adequate projectional sampling becomes tremendously more complicated. With the orientation of the bistatic system changing perhaps a great deal between pulses, the orderly progression of families of elliptical arcs from pulse to pulse is no longer present. Not only do these families of arcs change in their angular orientation relative to the ground patch, they also change with respect to the bistatic origin and in focal length. In addition, there is the potential problem with large increments in A under some conditions, as mentioned earlier and as is apparent for the vertical case shown in Fig. 5. 10. It is not difficult to imagine a situation in which some or all of these effects combine over the duration of several pulses so that part of the ground patch is effectively undersampled. The conditions for adequate sampling are an aspect of the bistatic SAR problem which were not studied here.

5.2.5 Attenuated propagation, simplified trajectories

This section represents one step forward and one step backward—reconstruction with the more physically representative condition of attenuation during propagation, but with the simplified trajectories. With this approach, it is possible to study the effects of attenuation without the added problem of trajectory effects.

The primary effect of propagation attenuation in SAR is to diminish the strength of the return signal from more distant reflectors, independently of their reflectivity. Generally speaking, if the radar is rather close to the ground patch so that the *variation* in propagation attenuation over the ground patch is significant, and if nothing is done to correct for it, then one would expect that the reconstructed image would suffer in its gray-level accuracy if not focusing.

The method of correction in monostatic SAR is clear; since all the scatterers which lie on equal-time-of-flight contours, the circular arcs, are affected in exactly the same way with respect to attenuation propagation, and since such reflectors contribute to exactly one point in a particular projection, it is necessary only to multiply that point by the inverse of the propagation attenuation. For example (see Fig. 4. 4), each point on the projection should be multiplied by $1/(R-p)^2$. The relationship between unattenuated and attenuated circular-arc projections is thus expressed exactly by this method, at least on this level of modeling. (To incorporate this into the simulations of Chapter 4 would be trivial and so was not done.)

Propagation attenuation in bistatic SAR is somewhat more complicated, as expressed in (5. 9), since each point along an elliptical arc is affected by a round-trip attenuation that is different from (almost) all of the other points on the same arc. This makes it impossible to make a correction by simply weighting each point of the projection by some number.

The approach used here to correct for attenuation propagation in bistatic SAR is based on the above correction for monostatic SAR, causing the backprojection process to be substantially modified. It is a generalization of the correction in monostatic SAR in somewhat the same sense that confocal coordinates are a generalization of polar coordinates ($F = 0$). Briefly, for each projection and each pixel, the attenuation function (5. 9) is computed along a hyperbola which passes through the pixel being computed for the backprojection. The attenuation is computed at the same values of B for which there are projection samples. The projection is then multiplied by the computed attenuation, convolved as usual, and interpolated to get the value to be added to the pixel for that backprojection.

The details of this reconstruction algorithm for bistatic SAR with attenuation propagation follow, except for the obvious conversions to discrete notation. These steps are repeated for each projection and each pixel at coordinates (x, y). First, the image coordinates (x, y) are converted to bistatic coordinates (x_b, y_b) via (5. 2). Then the hyperbola which is characterized by C and which passes through that point is found,

$$C = \frac{1}{2}\left[\sqrt{x_b^2 + (y_b + F)^2} - \sqrt{x_b^2 + (y_b - F)^2}\right],$$

and

$$\sin U = \frac{C}{F}$$

$$\cos U = \sqrt{1 - \sin^2 U}\ .$$

Since the values of B are known from the sampling points, the quantities

$$F \cosh V = B$$

$$F \sinh V = \sqrt{\cosh^2 V - 1}$$

can be found which correspond to the original samples of the projection—a set of ellipses which are orthogonal to the specified hyperbola. The necessary points for weighting along the hyperbola are found, in the manner of (5. 14), by

$$x_w = F \cos U \sinh V$$

$$y_w = F \sin U \cosh V\ .$$

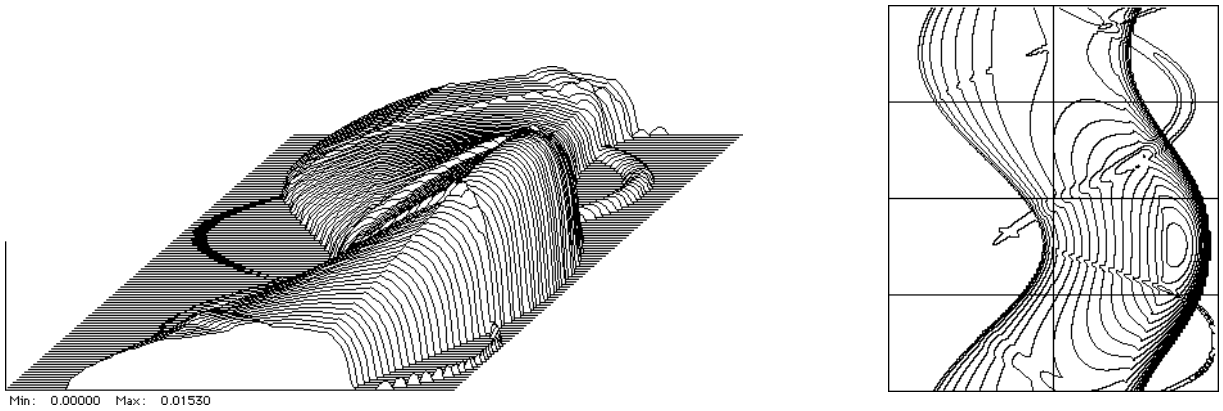
Each point in the projection is multiplied by the inverse of the attenuation function,

$$\sqrt{\left[x_w^2 + (F + y_w)^2\right]\left[x_w^2 + (F - y_w)^2\right]}\ . \qquad (5.\ 17)$$

The final steps are to convolve the new attenuation-weighted projection with the usual kernel and to interpolate (in B) between the resulting samples to find the contribution to the image pixel at (x, y).

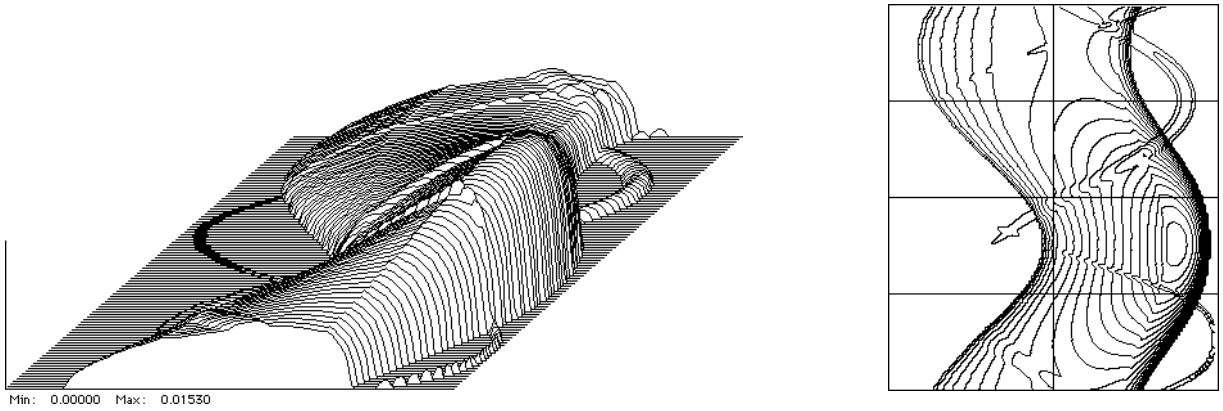
As described above, the algorithm may appear very to be computationally intensive. However, in spite of the many appearances of transcendental functions, it is not necessary to compute even one. Also, a simple observation saves it from possibly being impractical. Since the only points of the convolved weighted projection are those that are needed for interpolation, the last step, huge savings can be had if only the needed points are computed. Assuming that the entire convolution would be computed using an FFT and that only two points are needed (for linear interpolation), the savings had by computing those two points directly by the convolution sum easily exceed an order of magnitude for the numbers used in these simulations. In spite of this, the algorithm requires much more computation than any previous one. A further note on computational complexity follows, in Section 5.2.7.

Following the format used in presenting the results of unattenuated propagation, Fig. 5. 22, Fig. 5. 23, and Fig. 5. 24 show the elliptical-arc projections for the vertical, horizontal, and oblique cases, respectively. Characteristic of these plots is a greater variation in amplitude of the projection of an individual top hat for different positions of the bistatic system around the circle. Reconstructions using the above algorithm along with the weighting of (5. 15) in inverse square form are shown in Fig. 5. 25, Fig. 5. 26, and Fig. 5. 27.



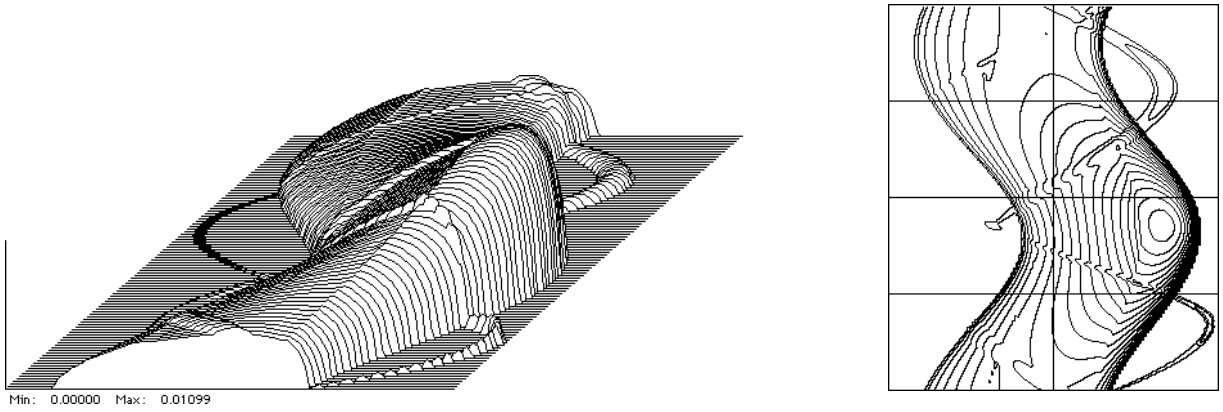
(a) (b)

Fig. 5. 22. Attenuated elliptical-arc projections for a horizontal initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



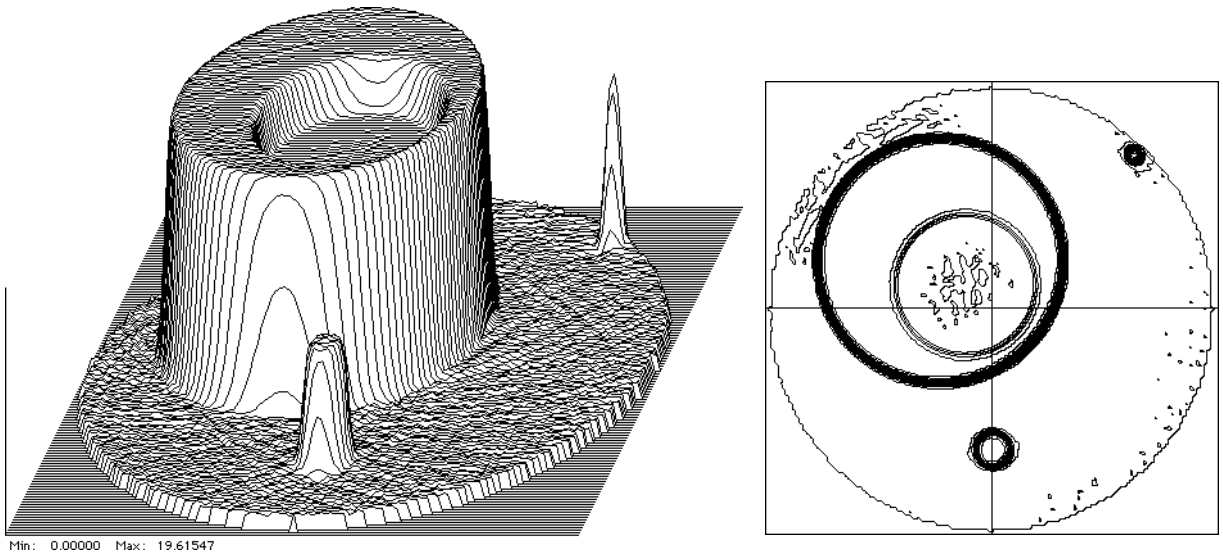
(a) (b)

Fig. 5. 23. Attenuated elliptical-arc projections for a vertical initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



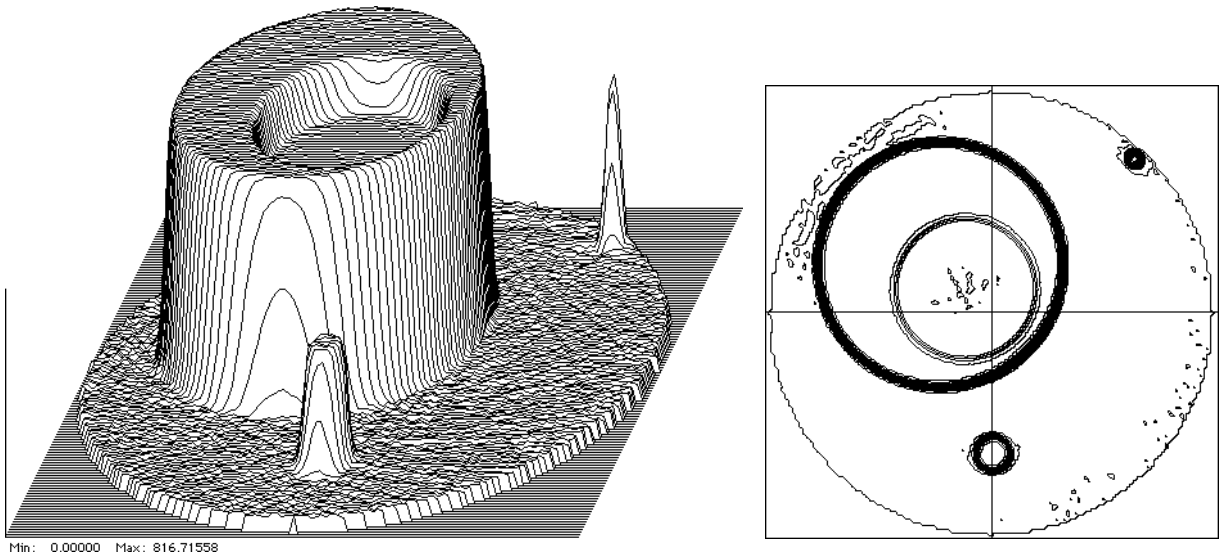
(a) (b)

Fig. 5. 24. Attenuated elliptical-arc projections for an oblique initial position of the bistatic system. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 25. Reconstruction from attenuated elliptical-arc projections, horizontal case, using modified elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.

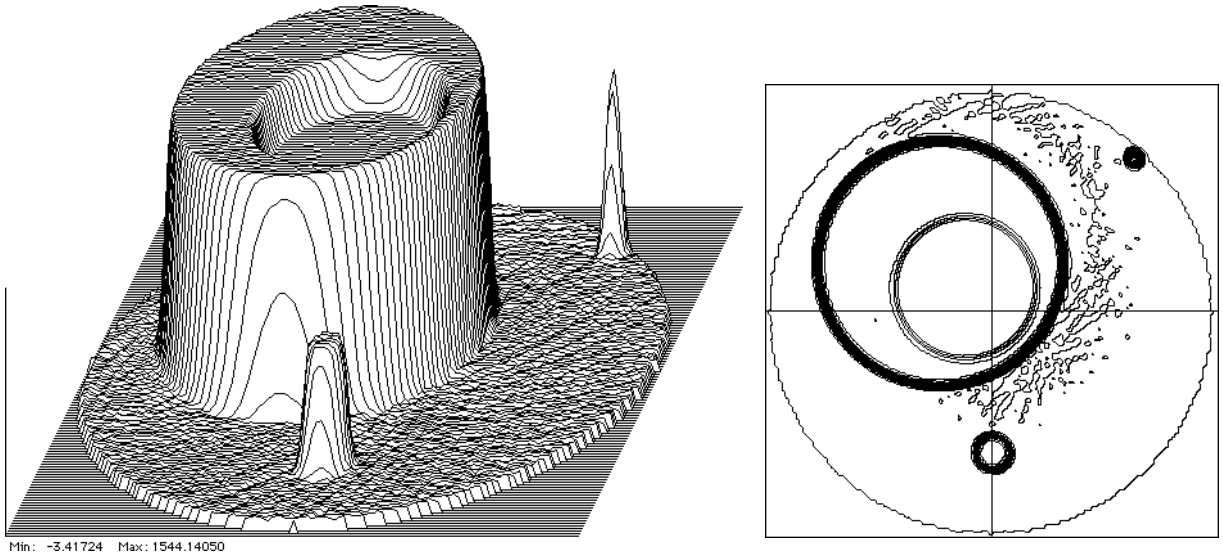


(a) (b)

Fig. 5. 26. Reconstruction from attenuated elliptical-arc projections, vertical case, using modified elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.

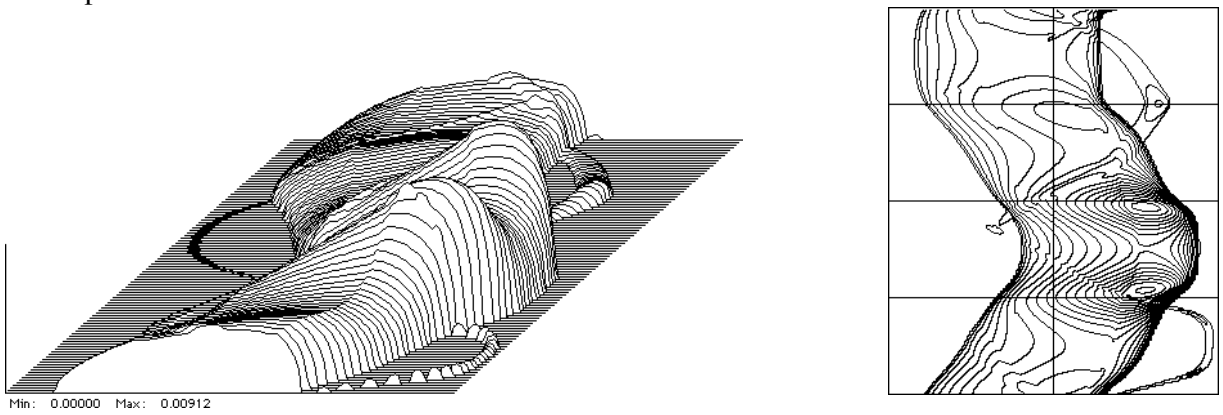
5.2.6 Attenuated propagation, extended trajectories

This final section requires little discussion. The algorithm used combines all of the elements of the earlier algorithms of this chapter. Stated another way, all of the earlier algorithms are special cases of this one. (Indeed, the same statement, applied carefully, is appropriate for the algorithms of Chapters 3 and 4 — Chapter 4 by letting $F \to 0$, Chapter 3 by letting $R \to \infty$.) The back-



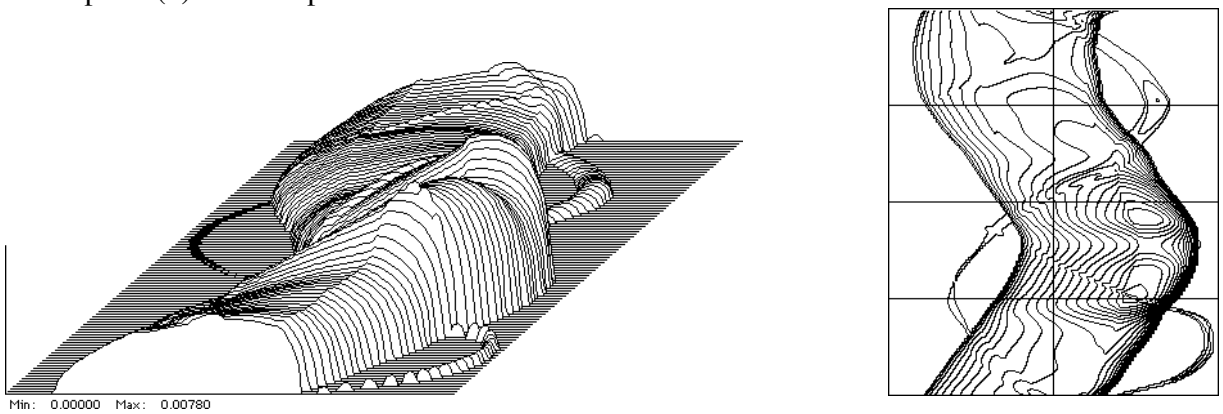
(a) (b)

Fig. 5. 27. Reconstruction from attenuated elliptical-arc projections, oblique case, using modified elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 28. Attenuated ellipitical-arc projections for the Type 1 trajectory. (a) Perspective plot. (b) Contour plot.



(a) (b)

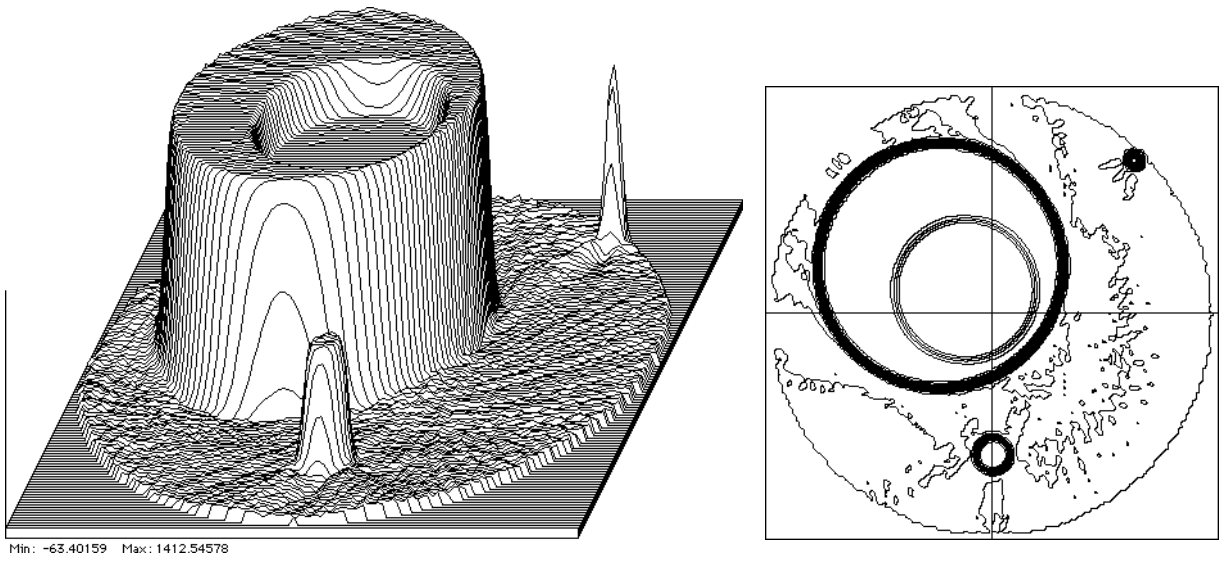
Fig. 5. 29. Attenuated ellipitical-arc projections for the Type 2 trajectory. (a) Perspective plot. (b) Contour plot.

projections are along elliptical arcs, there is a projection preweighting (5. 16), a pixel-by-pixel projection weighting (5. 17), and a backprojection weighting of the inverse square of (5. 15). Using the earlier designations for the extended trajectory types, Fig. 5. 28 and Fig. 5. 29 show the attenuated elliptical-arc projections for Type 1 and Type 2 trajectories, respectively. Fig. 5. 30 and Fig. 5. 31 show their respective reconstructions, which appear very similar to the reconstructions from unattenuated projections, Fig. 5. 20 and Fig. 5. 21.

5.2.7 Comments

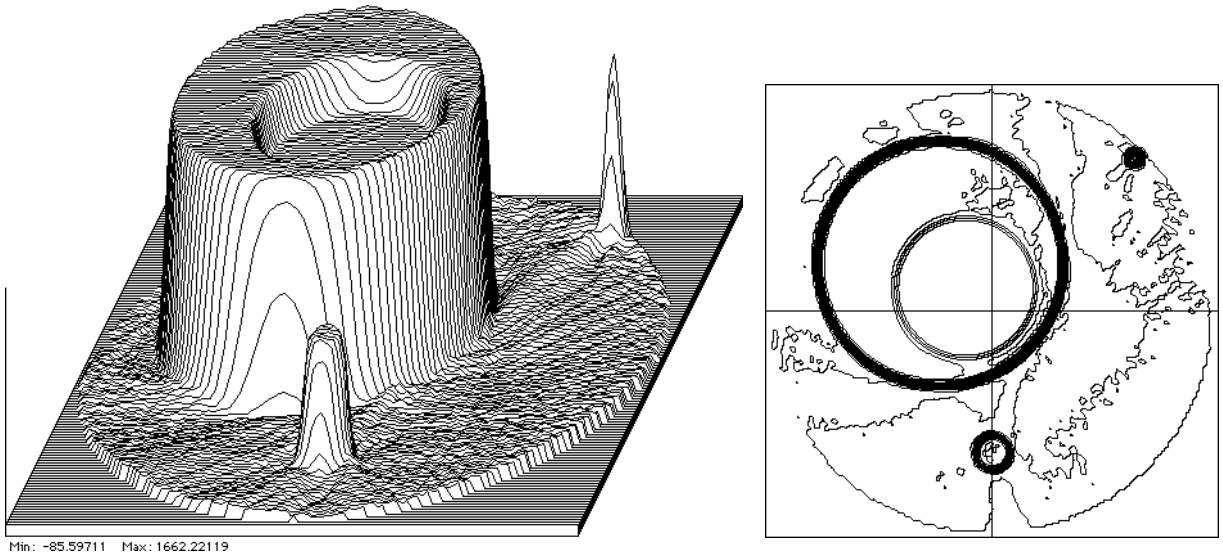
This chapter has dealt with bistatic SAR under some rather extreme geometries in order to highlight problems that arise due to wavefront curvature. A central issue was that of projectional sampling and how to correct for it. Perhaps the most desirable circumstance would be to have the sampling such that the monostatic case is generalized by having line integrals which cross the ground patch in equal increments along the hyperbolas of the confocal coordinates, that is, uniform sampling in u. However, this is inconsistent with the physical situation, which dictates integration along the confocal ellipses, with different u-increments effected at each point. It is the task of the reconstruction algorithm to correct for this and other phenomena.

Finally, running times for the two programs of this chapter will be quoted, as at the end of Chapter 4, Section 4.5. The program that reconstructs without correcting for propagation attenuation is called EACBPU, for Elliptical-Arc Convolution Back-Projection, Unattenuated, and it runs in 21.0 min. The program that reconstructs with correction for propagation attenuation is called EACBPA. Its four levels of looping, instead of the three levels used in all other reconstruction programs in this dissertation, cause it to run for 13.78 h.



(a) (b)

Fig. 5. 30. Reconstruction from attenuated elliptical-arc projections, Type 1 trajectory, using modified elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.



(a) (b)

Fig. 5. 31. Reconstruction from attenuated elliptical-arc projections, Type 2 trajectory, using modified elliptical-arc convolution backprojection. (a) Perspective plot. (b) Contour plot.

CHAPTER 6

RELATED TOPICS AND FURTHER RESEARCH

THE PURPOSE of this chapter is to present several topics which are related in some fashion to the material of the preceding chapters. The topics can be loosely classified as extensions of the earlier theory which can be posed as tomographic problems, more or less obvious areas for further investigation, and a small collection of what are believed to be novel radar imaging techniques. The amount of work that has been done from topic to topic varies widely; some topics have been studied through the early stages of simulation and some are only ideas. In spite of this, the problems in this chapter are offered in the hope that they will be found to be as important a contribution as the chapters which precede it.

6.1 Antenna Shading of the Ground Patch

With the wide-angle imaging that is now possible using the algorithms of Chapters 4 and 5, another problem presents itself; the effects of nonuniform shading of the ground patch by the antenna pattern[10] are exacerbated. When a narrow-angle patch is being imaged, the antenna normally illuminates a much larger piece of ground and presumming is used to remove the unneeded information. This process usually justifies the assumption that the antenna is omnidirectional, since the variation in pattern over the imaged patch is relatively small. With the ability to image a wider patch, this procedure should be reexamined since the projections are now weighted along the line of integration. In terms of algorithm development, this "transverse" weighting is quite different than the weighting caused by propagation attenuation. This section will describe the problem and indicate possible solution methods. It is noted that some current systems employ other approaches to this problem.

Two versions of the antenna shading problem will be shown: the plane-wave case, in which Fourier methods such as the Projection-Slice Theorem can be brought to bear as needed, and the case of circular-arc projections. Both are sources of insight, but the former is limited in the range of physical situations to which it applies, and the latter is limited in its range of analytical tools.

For the plane-wave situation, Fig. 3. 2 will once again be helpful. The antenna pattern which is superimposed on the ground patch will rotate with the circling radar, so for each look, it is only a function of y'. Since it is necessary, for this problem, to model only the main lobe (one should expect major problems in the reconstruction if a null in the antenna pattern falls in the ground patch),

[10] Unless stated otherwise, monostatic SAR will be assumed hereafter.

a cosine function should suffice as an approximation. (A cos θ function closely resembles a slightly fattened sin $2\theta/2\theta$ function between $-\pi/2$ and $+\pi/2$. Such a pattern might result, for example, from a tapered linear antenna.) Therefore, the antenna pattern

$$a(y') = \cos\left(\frac{\gamma\pi}{2L}y'\right)$$

seems useful. With $\gamma = 1$, the first null falls exactly at the edge of the ground patch. A value of $\gamma = 0.9$ would therefore seem safe and in keeping with the policy of using stringent test conditions for the simulations. (An observation made in passing is the fact that this pattern appears as only a function of y' which implies that it is a backprojection with respect to x'.) This model of the antenna pattern also assumes that it is independent of frequency, which is not a realistic assumption for some cases. To eliminate this assumption would make an even more interesting problem.

The calculation of the antenna-weighted straight-line projections of a top hat is straightforward, being simply

$$f_{la}(p, \Theta) = \int_{y_1'}^{y_2'} a(y')\,dy' .$$

The values for the y'-projected top hat intercepts, y_1' and y_2', for a top hat of radius a and rotated coordinates $\left(x_0', y_0'\right)$, are

$$y_{1,2}' = y_0' \pm \left[a^2 - \left(x' - x_0'\right)^2\right]^{1/2} .$$

The integral then evaluates to

$$f_{la}(p, \Theta) = \begin{cases} \dfrac{4L}{\gamma\pi}\cos\left(\dfrac{\gamma\pi}{2L}y_0'\right)\sin\left\{\dfrac{\gamma\pi}{2L}\left[a^2 - \left(p - x_0'\right)^2\right]^{1/2}\right\}, & \left|p - x_0'\right| < a \\ 0 , & \text{otherwise} \end{cases}$$

in which

$$x_0' = x_0\cos\Theta + y_0\sin\Theta$$

$$y_0' = -x_0\sin\Theta + y_0\cos\Theta .$$

To clarify what is meant by antenna-weighted projections, Fig. 6. 1 shows such projections for the single, narrow top hat of the test function alone.

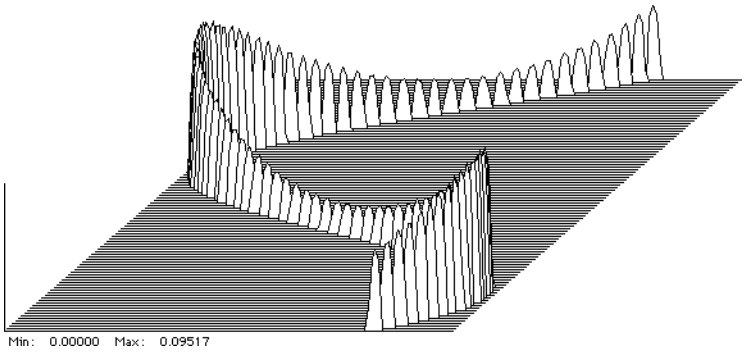


Fig. 6. 1. Antenna-weighted straight-line projections of the narrow top hat of the test function.

Several attempts were made to reconstruct the test function by doing obvious things, but with little success. Weighting the backprojections according to the antenna pattern or its inverse is not the correct procedure. It was learned that performing a point-by-point reconstruction (see Chapter 2) with the antenna weighting along the integration paths, in matched-filter style, gives exactly the same result as weighting the backprojections with the antenna pattern—it is necessary only to reorder the loops in the program. Other methods were tried and much was learned about the problem, but apparently this seemingly "textbook" problem is more subtle than it first seems. One observation that may be worthwhile is that since the antenna shading rotates with the radar from look to look, the image measured each time is different, effectively making it a time-varying image. Perhaps a fruitful avenue of investigation would cast the problem in a time-varying signal processing framework. The following discussion may cast more light on the subject.

As is well known, the Projection-Slice Theorem is a powerful tool in many tomographic problems. The antenna-shading problem affords an opportunity to generalize this theorem, although its utility here is not well-established. Consider an image $f(x, y)$ which is being shaded (multiplied) by a function $a(y)$ which has an inverse Fourier transform

$$A(\alpha) = \frac{1}{2\pi}\int a(y)\exp(j\alpha y)\,dy\,.$$

For the moment, the radar will be assumed to be at $\Theta = 0$, so that the notation $a(y)$ is consistent with the earlier usage $a(y')$. Then the projection of the shaded function is

$$f_l(x, 0) = \int f(x, y)\,a(y)\,dy$$

$$= \frac{1}{(2\pi)^2}\iiint F(X, Y)\exp(jxX + jyY)\,a(y)\,dy\,dX\,dY$$

$$= \frac{1}{(2\pi)^2} \int\int F(X, Y) \exp(jxX) \left[\int a(y) \exp(jyY)\, dy \right] dX\, dY$$

$$= \frac{1}{2\pi} \int\int F(X, Y) \exp(jxX)\, dX\, A(Y)\, dY .$$

Taking the Fourier transform with respect to x of each side results in

$$\mathsf{F}_x\{f_l(x, 0)\} = \frac{1}{2\pi} \int F(X, Y)\, A(Y)\, dY .$$

This result easily generalizes to the case $\Theta \neq 0$ where $a(y')$ is used. The details are left out, but are based on the fact that rotation of an image implies the same rotation of its two-dimensional Fourier transform. The result, in words, is: The one-dimensional Fourier transform of a Θ-projection of an a-weighted function $f(x, y)$ is the Θ-projection of the A-weighted function $F(x, y)$. This generalizes the Projection-Slice Theorem because that theorem follows with $a(y) = 1$.

The new result may be useful in working on the antenna shading problem, but on the surface it appears to highlight the difficulty of the problem since it states that there is nothing to be gained by working in the Fourier domain. The problem simply reappears in the Fourier domain in the same form as in the spatial domain—find a function from projections under the influence of a rotating one-dimensional shading. The effect of the shading function, in the Fourier domain, is to smear out the "slice" that is so useful in the Projection-Slice Theorem.

A specialized version of the above result appears in a paper by Farhat [73]. It can be derived by letting $a(y) = \exp(j\alpha y)$, so that

$$f_{le}(x, 0) = \int f(x, y) \exp(j\alpha y)\, dy$$

resulting in

$$\mathsf{F}_x\{f_{le}(x, 0)\} = \frac{1}{2\pi} F(X, \alpha),$$

an offset slice. If the sidelobes of the antenna are low enough to allow the approximation that the ground patch is zero outside the main beam, and if the cosine is a good enough approximation to the main beam, then this affords some insight to the severity of the problem. With the cosine main beam and the identity $\cos(\theta) = (\exp j\theta + \exp -j\theta)/2$, it is clear that instead of retrieving a central slice of the Fourier transform of the image, what is retrieved is the sum of two offset slices. It should be possible under these conditions to determine the amount of offset for a particular system design and to determine if it is enough to cause significant image degradation. If it is deemed necessary to try to repair the damage, one possible method would mimic coherent Doppler processing,

by borrowing from monopulse tracking radar. In Doppler processing, it is possible to measure the frequency offset of a reflected sine wave in absolute value easily; however, it is impossible to decide if the reflector is approaching or receding. The spectrum of the baseband Doppler signal is symmetric about the origin. The ambiguity can be resolved by transmitting simultaneously a cosine signal (or more practically, by synthesizing it in the receiver), forming a quadrature channel. This effectively nulls out either the left-side or the right-side impulse of the original Doppler spectrum depending on whether the reflector is approaching or receding. In monopulse tracking radar, the same concept is implemented using two antenna patterns. One pattern, called the "sum" pattern, is an ordinary, cosine-like lobe. With this, an object can be detected but tracking is impossible because it can not be known whether the object is to the left or to the right of boresight. With a second pattern having a null at boresight and a positive and a negative lobe to either side, the actual direction can be determined from the sign of the ratio, for example. This second pattern resembles a sine shape, and is called the "difference" pattern. (The nomenclature "sum" and "difference" relates to forming the two beams by respectively adding and subtracting the outputs of the elements of a two-element antenna.) In the SAR situation, the addition of a sine pattern would eliminate one of the offset slices so that the data collected would represent only a single offset slice. Inversion could be accomplished by any suitable direct Fourier inversion. One consequence of the offset samples is that the origin would not be sampled, possibly giving rise to other reconstruction artifacts. Of course, synthesizing these patterns can not be done exactly, so such endeavors would be towards improving the image quality without necessarily fixing it completely.

Another way of repairing the problem (if in fact it needs repair) would focus on antenna design in another sense. If an antenna could be designed that had a relatively constant-valued transmission over its main beam (a brick-wall antenna, so to speak), then the problem would be solved. It may pay to coherently process a few return signals with the sole intention of synthesizing such an antenna, but leaving the SAR imaging phenomenon conceptually intact. Such a method might be thought of as a sub-aperture "real-beam" synthesis. Of course, spatial sample spacing would be an important issue, just as in other arrays.

It is also useful to investigate the antenna shading problem under the spherical-wave assumption, bringing all the aspects of the far field (4. 4) into play. Referring to Fig. 4. 3 and Fig. 4. 4, let the antenna always point along the negative x'-axis with θ measured from the same axis. A suitable cosine-style antenna pattern in the manner of that used previously is

$$a(\theta) = \cos\left(\frac{\gamma\pi}{\theta_M}\theta\right)$$

where $\theta_M / 2$ measures the angular extent, from the negative x'-axis, of the ground patch as seen from the radar, that is,

$$\frac{\theta_M}{2} = \sin^{-1}\frac{L}{R}.$$

A value of $\gamma = 1$ brings the first null to the edge of the ground patch. The projections of a top hat under this shading can be calculated by integrating the antenna pattern along a circular arc representing a wavefront between the limits given by (4.5), giving

$$f_{ca}(p, R, \Theta) = \begin{cases} \dfrac{r\theta_M}{\gamma\pi}\left[\sin\left(\dfrac{\gamma\pi}{\theta_M}\theta_2\right) - \sin\left(\dfrac{\gamma\pi}{\theta_M}\theta_1\right)\right] & \text{if } -1 \le \dfrac{\hat{r}_0^2 + r^2 - a_0^2}{2r_0 r} \le 1 \\ 0 & \text{otherwise} \end{cases}$$

in which the intermediate variables are defined in the earlier derivation of unshaded circular-arc projections. The projections of the narrowest top hat of the test function resemble those of Fig. 6. 1 except that the contours tend along the distorted sinusoidal pattern that is characteristic of the circular-arc projections of Chapter 4.

With the spherical-wave formulation of the problem and the angular spectrum interpretation of the far field at hand, a possible reconstruction method suggests itself. Since the plane wave spectrum states that the far field is a sum of plane waves with varying weights, and since reconstruction from straight-line projections is known, it should be possible to decompose the spherical wave into a set of plane waves with known weights. Then each circular-arc projection would be backprojected along linear paths multiple times, but each time the direction of backprojection would be offset from that for the nominal (boresight) direction and weighted according to the corresponding elemental plane wave. Of course, the process would have to be discretized for computer implementation. As described, the increase in computation over that required for ordinary straight-line convolution-backprojection is roughly the number of plane waves which are used to approximate the spherical wave. However, since the algorithm designer has freedom over how the plane-wave spectrum is to be discretized, it could be done so that a kind of rebinning, that is, combining projections from different looks which are to be backprojected in the same direction *before* the backprojection instead of after backprojection, could be implemented to reduce computation.

6.2 Platform Motion, Ground Patch Motion, and Intrapulse Doppler

Other "second-order" effects which should be re-examined in light of the wide-angle imaging ability include those relating to relative motion between the radar and the ground patch. These motions were simplified in the "stop-and-go" model in order to facilitate analysis and algorithm development. This section comprises some qualitative preliminary thoughts on this subject. By and large, the stop-and-go model remains highly useful; it appears that correction for these motion

effects would be beneficial only in systems requiring very high resolution and probably with other extreme system requirements such as the high velocities of low Earth orbits and the wide bandwidth and extended pulse duration associated with low probability of intercept (LPI) radars [20], [89]. The range of resolutions affected may be so small that other factors may contribute more deleterious effects than uncorrected motion. However, with higher demands being placed on new imaging radars, each system would have to be analyzed in detail in order to decide if such correction is needed. The remainder of this section assumes plane waves. Extensions to curved waves are mostly obvious at this level of discussion.

The first case to be considered is that in which the ground patch is considered to be stationary and the radar is stationary while transmitting and receiving, but moves between those times. This is closer to reality (in many instances) than the stop-and-go model. It matters not whether the transmitted signal is an impulse or some other signal. This is clearly the same situation as the bistatic SAR studied in Chapter 5 in which the transmitter and receiver were assumed to be stationary while transmitting and receiving. If this motion model is adequate for the application and correction is necessary, it can be achieved by a suitable bistatic imaging algorithm.

The next step up in increased sophistication has a continuous motion between the radar and the ground patch but still transmitting an impulse. Even though it makes no difference whether the ground patch is considered stationary and the radar in motion or vice versa (or whether some other frame of reference is used), it is sometimes helpful to be able to visualize the motion either way. For Earth satellites, a full accounting of relative motion would include both the radar's trajectory above the Earth and the rotation of the Earth below the radar in a vector sum, relative to an inertial reference. In any event, it is possible to retain the two-dimensional (zero elevation) model used so far without losing the salient features of relative motion. The effect of a non-zero depression angle is to reduce the apparent rotation rate of the ground patch relative to the radar. The amount of reduction would depend on the squint angle of the antenna, but a safe estimate of that reduction would surely be to reduce the effective radar velocity to 8500 mi/h, about half that of the linear velocity of a satellite in low Earth orbit. At least on an instantaneous basis, the two-dimensional scenario has the radar traveling around the ground patch at a speed of 8500 mi/h, in the plane of the ground patch. It can be shown (see Appendix C) that the projections are no longer along parallel straight lines but along non-parallel straight lines which resemble the fan rays in fan-beam tomography. The analysis shows that these projection rays do not form an actual fan because they do not have a common center in general. However, it can also be shown that under almost any condition that is likely to be encountered, the rays do form an approximate fan. Simple calculations based on Earth orbit show that the error in backprojecting along parallel straight lines relative to backprojecting along the fan rays is probably negligible unless the desired resolution approaches one centimeter. These calculations used a range R of 200 mi, an antenna beamwidth of six degrees, and the reduced linear velocity (to satisfy the two-dimensional scenario) of 8500 mi/h. The effec-

tive rotation rate of the ground patch relative to the radar Ω is 42.5 rad/h and the patch radius L is 10.5 mi. The maximum distance between the fan rays and the parallel rays over the ground patch can be found to be

$$\varepsilon_{max} \approx \frac{L \sin\left(\frac{\Omega L}{c}\right) + L}{\cos\left(\frac{\Omega L}{c}\right)} - L \approx \frac{\Omega L^2}{c} \text{ if } \frac{\Omega L}{c} \ll 1$$

which leads to the claim that the fan beam effect is negligible unless the resolution is on the order of a centimeter. Other system parameters will give different results, of course. It is to be remembered that this calculation assumes a transmitted impulse, the signal which minimizes the fan beam effect. The next paragraph derives a more restrictive result for the case of extended pulses.

The final step in the modeling of motion is to allow continuous movement as before and also to allow an extended pulse to be transmitted. In the previous paragraph, the signal that was reflected from the ground patch due to the transmitted impulse lasted $2L/c = 113$ μs, and during this time a very slight fan beam effect was observed. To understand how an extended pulse is reflected from the ground, it is helpful to think of the transmitted pulse as being approximately composed of a series of very short pulses (pulse segments) each of which is short enough that it can be analyzed in the same way as a transmitted impulse. Then the return signal consists of a series of fan beam projections. Most likely, the fan effect is very slight, as before, and does not require correction. Each fan beam is rotated very slightly from the previous one due to the later transmission time of that pulse segment. Therefore, an appropriate question to ask is, how much is the final fan beam rotated from the first one. Even if the fan effect does not need correction, it may still be necessary to correct for this intrapulse rotation. The same system parameters will be used as were used in the previous paragraph, with the addition of a $T_p = 1000$ μs pulse as may be typical of a LPI SAR. [11] The amount of rotation of the ground patch relative to the radar during the total 1113 μs return signal is 0.000013 rad. A point on the y'-axis at L will experience the greatest amount of range error if the same straight-line backprojection is executed for the entire return signal. This error, using the 10.5 mi patch radius, is 8.7 in or 0.22 m. This amount of range error can be expected to affect resolution with resolution cells on the order of three feet or one meter. Although this is in the realm of high resolution, it may behoove the designer of future systems to be aware of this limitation.

A summary of the discussion of this section so far is in order. As more detail was added to the model for relative motion between radar and ground patch during data collection, three distinct

[11] Pulse lengths for LPI radars are hard to discover, at least in the open literature. The value of 1000 μs seems to tend towards the longer range of some radars which are known. Figure 6-2 of [89] shows a "scattergram" of pulse duration *versus* bandwidth of a number of radar designs. Also, [21] proposes an imaging radar for space object identification which would use a 1 ms pulse and a 1 GHz bandwidth centered at 94 GHz for $TB = 106$. Another proposal is [61] which would have a 2 ms pulse and a 1 GHz bandwidth at X-band, $TB = 2 \times 106$.

methods for correction were proposed, along with the observation that the stop-and-go model is usually adequate, even with the ability to image larger patches and using longer pulses for LPI. The first method is to use bistatic reconstruction when the distance traveled between transmission and reception is significant. The second method is to use fan-beam reconstruction when the patch size is long enough in its range extent that significant platform motion happens during reception of the reflections causing a fan beam effect. The third method, appropriate for very long pulses, is to separately backproject the received signal in pieces along different rays, just as if the transmitted signal consisted of a series of short pulse segments. Here, the backprojections can be either the fan type or the straight-line type, as needed. One can ponder the addition of wavefront curvature to these problems. For example, the fan beam backprojection would be adapted so that the rays of the fan become circular arcs.

The final topic to be discussed in this section is that of intrapulse Doppler and its variation over the cross-range dimension of the ground patch. In a way, the preceding discussion of fan beams lends insight here, but there is another way of looking at this that is also beneficial. Under the same motion conditions as the previous paragraph, it is clear that points at different values of y' will cause different amounts of Doppler shift to be imposed on the pulse. With the convention that the (x', y') system (fixed on the radar) rotates in a counterclockwise direction relative to the (x, y) system, points on the positive y'-axis will reflect with a positive Doppler shift and the amount of shift is proportional to y'. A similar statement applies to the negative y'-axis and negative Doppler shifts. If the amount of Doppler shift were independent of y', it would be a simple matter to retune the matched filter so that it is matched to the Doppler-shifted signal if it was found that the output of the original matched filter was reduced too much. However, with the situation as it stands, the receiver would have to undergo a major redesign. It therefore makes sense to find the amount of Doppler that can be accommodated without signal degradation.

The above calculation has been done by Rihaczek [18], but only in the context of a single Doppler shift affecting the entire signal. The result, stated below, assumes that the complex envelope of the signal is simply delayed and not distorted, and that only the phase part of the signal is affected by Doppler. It is stated that a phase shift of π radians over the signal duration causes the peak of a matched-filter output to be reduced by about 3 dB from its matched condition; therefore, the derivation continues by allowing only $\pi/2$ radians over the pulse duration. The result for negligible signal loss is then given as

$$TB \lesssim 0.1 \frac{c}{\left|\frac{dR}{dt}\right|} \tag{6. 1}$$

where TB is the time-bandwidth product of the transmitted pulse. The sensitivity to the higher-order range derivatives is said to be less than that for the first derivative.

The application of the above constraint to SAR is as follows. Presumably, the receiver will be matched for points at the center of the ground patch, i.e., for zero Doppler. If $|dR/dt|$ for reflectors located at the larger values of $|y'|$ exceeds the constraint, then their reflectivity will not be properly measured. The amount of reduction in signal strength due to filter mismatch will in general not be linear with y' but will depend on the details of the waveform. The result is a transverse weighting of the projections not unlike that caused by antenna shading, as discussed in the previous section. Another factor to be considered is that the output of a mismatched filter surely has distortions other than merely a reduced peak. These distortions may contribute significantly and directly to a loss in resolution. This issue is not addressed in the derivation of (6. 1).

It remains to show whether (6. 1) constrains a spaceborne SAR. Using the same system parameters used earlier, that is, $\Omega = 42.5$ rad/h and $L = 10.5$ mi, the range rate for a point at L on the y'-axis is $|dR/dt| = \Omega L = 445$ mi/h, constraining TB to be less than 150,000. For a range resolution ΔR, the usual relationship to bandwidth B is $\Delta R \approx c/2B$. For $\Delta R = 1$ m, this implies a bandwidth of 150 MHz. Using the earlier LPI-related value of pulse duration $T = 1000$ μs gives a time-bandwidth product of 150,000. This value is marginal with respect to the constraint, so it appears that the problem of image degradation due to intrapulse Doppler should not be ignored for some spaceborne SARs.

A correction for the intrapulse Doppler might have a bank of matched filters each tuned to the center of a resolution cell measured along y'. A tomographic method based on correcting the transverse weighting would be an alternative and may well be shown to be equivalent.

This section has considered several "second-order" effects which in large part are aggravated by attempts to image a larger ground patch, as measured by subtended angle. It is believed that these problems have not been addressed (or perhaps even identified) in the literature. Simple calculations show that although some are generally truly small effects, they can not all be ignored in some applications. While other solution methods are no doubt available, the tomographic methods outlined here seem appropriate; indeed, the tomographic framework aids the identification and interpretation of the problems.

6.3 Other Imaging Methods and Modes

This section will briefly introduce several possibilities for radar imaging. Some are mutations of the imaging styles discussed in this dissertation and in other literature and some are more novel. The issues of applications and practicality will not be addressed here. The author has not seen these broached in the literature and therefore believes them to be new.

6.3.1 Exploiting wavefront curvature

With the constraining plane-wave assumption lifted, it might become possible to exploit wavefront curvature rather than to merely tolerate it. Two examples of such will be be mentioned

here—there may well be others. First, imagine a bistatic radar at some distance from the ground patch. The transmitter and receiver revolve around a common point, traversing circular trajectories, as in Fig. 6. 2 (a), so that the origin of the bistatic coordinates is stationary. Under the plane wave assumption, the effective angle of the projections is constant because the bisector of the bistatic angle is fixed. With circular waves and elliptical contours of equal-times-of-flight, the local tangent to a contour changes at points on the ground patch, for different intersecting ellipses. This effect is difficult to portray, but the crossed straight lines in Fig. 6. 2 (a) are drawn to be tangent to two such ellipses of equal eccentricity which happen to intersect at the same point. Obviously, there are ellipses of differing eccentricities which intersect the point for all positions of the transmitter and receiver, but these are hard to draw correctly and the alternate display is used. An expression that is useful in determining the amount that the elliptical contour deviates in tangency relative to a plane-wave contour can be found by subtracting the angle of the tangent to a circle centered at the bistatic origin from the angle of the tangent of the ellipse. The circle is useful because at any point to be examined, its tangent has the same slope as would a straight line used in a plane-wave approximation. Shown in Fig. 6. 2 (b) is the construction for the calculation, which results in

$$\zeta = \arctan\left(-\frac{B^2}{A^2\tan\theta}\right) - \arctan\left(-\frac{1}{\tan\theta}\right).$$

The maximum variation in this curvature-induced look angle is found by setting $d\zeta/d\theta$ to zero, or

$$0 = \frac{B^4}{A^4}\cos^2\theta + \sin^2\theta - \frac{B^2}{A^2},$$

solutions of which can be found numerically. If a range of A/B for a particular application were known, this would be useful in finding a nominal orientation of the bistatic system in order to maximize the variation in ζ if that should be necessary. This imaging scheme also exploits the phenomenon of speckle imaging, since only a few degrees of variation in ζ would normally be available. In fact, the speckle aspect is probably needed for this to work at all.

The second example in which wavefront curvature might be used beneficially is depicted in Fig. 6. 3. A monostatic radar flies in a straight line. As is commonly understood, it is not possible to obtain an image directly in front of the radar platform, along the line of travel. The tomographic explanation of this is that there is no angular variation of the wavefront in that direction. This is true regardless of whether plane waves or spherical waves are thought to be present. The spherical-wave case is illustrated in Fig. 6. 3 (a). However, with spherical waves, one feels that points somewhat displaced from the flight path would experience some angular diversity strictly due to what might be called “radius modulation,” caused by the upper and lower branches of the circular

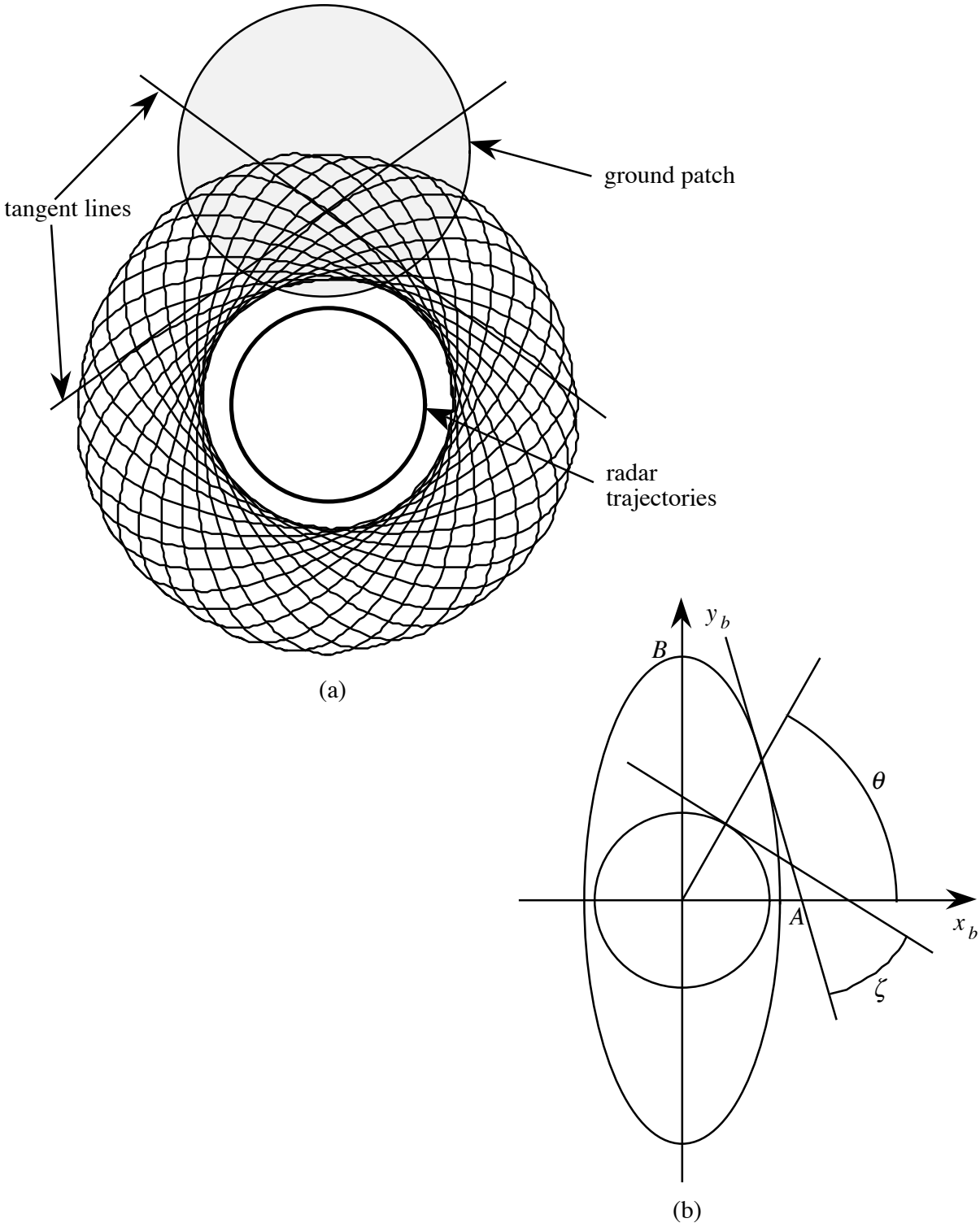


Fig. 6. 2. (a) Geometry for SAR imaging from a bistatic system with a stationary origin. Several elliptical contours of equal times-of-flight are shown. (b) Geometry for deriving the variation in tangency for elliptical contours relative to the straight-line contours which would result from plane waves.

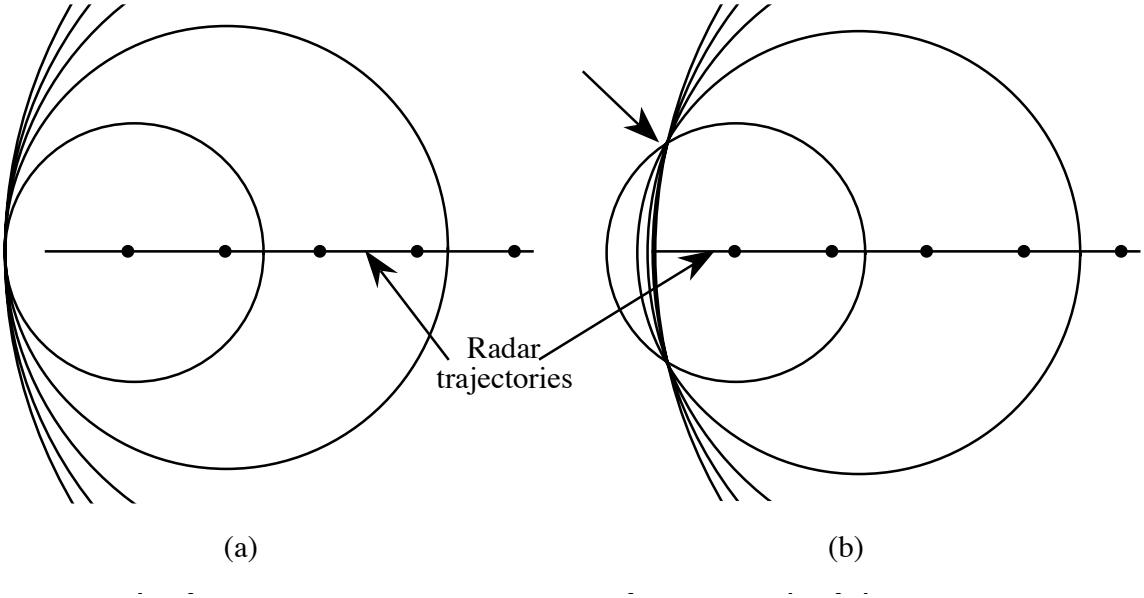


Fig. 6. 3. Imaging in forward directions with linear flight trajectories. (a) No image is possible directly in front of the radar. (b) Not predicted by plane-wave theory, an image can be formed at points away from the flight path.

arcs being "pulled in" as the radar approaches. This effect is illustrated more clearly in Fig. 6. 3 (b) by showing the circles offset along the flight path by just such an amount that they intersect at a point off the flight path. (The same effect could be made by keeping the circle centers at the same place as in Fig. 6. 3 (a) and adjusting the radii somewhat.) The circular arcs intersect this point with a variety of slopes. Again, this type of imaging would be enhanced by the speckle phenomenon. Other scenarios can be envisioned in which imaging areas displaced from the nominal squint angle are enhanced when the squint angle is not straight ahead. Here, the effect of radius modulation adds to the normal effect of platform motion that is normally used in image formation under the plane wave assumption. Notice that it is always possible to select one point, such as the one indicated by the unlabeled arrow in Fig. 6. 3 (b), and draw tangents to the intersecting circles which could represent plane waves. With that particular reference direction chosen for the plane waves, for that particular point, and for each pulse, there is no difference. The wavefront curvature can be beneficial only for points which are positioned *away* from the wave propagation vector.

6.3.2 Variations on rho-filtered layergrams

What appears to be a promising method of deriving reconstruction algorithms for the more unusual geometries will now be described. Essentially, the method and the algorithms so derived would mimic what is called the rho-filtered layergram or filter of backprojections. The equations of conventional tomographic reconstruction are often manipulated (usually using the Projection-

Slice Theorem) so that instead of indicating that a one-dimensional filtering operation should be applied to the projections before they are backprojected, they indicate that an equivalent operation is to first backproject the projections and then apply a two-dimensional filtering operation. In [60], an alternate derivation is given (if concisely) which does not depend on the Projection-Slice Theorem. Briefly, from (2. 8) and (2. 9), the backprojection of all of the projections of an impulse at the origin is

$$h(r, \phi) = \frac{1}{\pi} \int_0^{\pi} \delta[r \cos(\theta - \phi)] \, d\theta$$

where the polar coordinates (r, ϕ) are used for the reconstruction plane. Using the identity

$$\delta[f(x)] = \delta(x - x_0) \left| \frac{df}{dx} \right|^{-1}_{x = x_0}$$

where $f(x_0) = 0$, the integral evaluates to

$$h(r, \phi) = \frac{1}{\pi |r|} \, . \tag{6. 2}$$

Assuming shift-invariance (which condition exists but is not shown here), the above represents the impulse response or point spread function of the backprojection operation. Since $|\rho|^{-1}$ is the Fourier transform of $(\pi |r|)^{-1}$, with r the polar variable in the Fourier transform plane, an image reconstructed using backprojection can be focused by multiplying its Fourier transform with $|\rho|$ and inverse transforming.

The above derivation may be useful when applied to some of the situations described in this dissertation. If not, a numerical procedure which emulates it could be used. For example, it would probably be relatively straightforward to use the above identity in the proposed reconstruction integral for circular-arc projection data when the flight path is circular. It is conjectured that the rho-filtering operation could be shown to be correct when the flight path is circular in monostatic SAR. However, it is also conjectured that such filtering is not correct when the flight path is square, as in Fig. 4. 18, and may account for some of the artifacts in that figure. It is not known whether similar success could be expected when applying the identity to other cases, such as the antenna shading problem of Section 6.1 or any of the bistatic cases. In these instances, the analysis may become intractable so that a numerical procedure could be undertaken. The procedure would be to first backproject a large number of unfiltered impulse projections along the appropriate path, such as elliptical arcs, in order to find an estimate of the impulse response analogous to (6. 2), from which a correction filter could be readily found. The existence of shift-invariance would have to be carefully examined in all cases by computing the impulse response for a number of impulse locations. For

example, in the antenna shading problem, the exact nature of the modulation (e.g ., modulation depth) that is experienced by a point reflector depends on its distance from the center of the ground patch; it might be expected that the correcting filter would then depend on that distance as well. In general, although the entire process would most likely not be practical to implement in real time, there may well be instances in which the computation could be done off line. The programs in Appendix D could be modified to perform this numerical task. Modifications would include shifting the beginning of sampling on each new look angle to ensure that a projectional sample would pass through the impulse location so that the impulse is sampled each time. Of course, bandwidth issues would also have to be addressed to make sure that the Fourier transformation result is valid.

6.3.3 Matched filtering

As mentioned in preceding chapters, matched filter concepts may prove useful in deriving new algorithms. Here, the usage of the term "matched filter" is in the vein of the discussions of [60] and [18] which, although their approaches are quite different, are particularly mindful of the fact that matched filtering can be more general than the usual autocorrelation process, taking the form of suitable inner products between the signal being examined and an ensemble of prototype functions which differ in one parameter [90]. If this parameter is signal delay, then the autocorrelation results; if the signal is a sine wave (or sum of sine waves) and the parameter is frequency, then the Fourier transform results.

The interpretation of [60] is couched in the context of X-ray tomography. The approach is to find the impulse response of the data collection machinery. For straight-line projections, this impulse response traces a sinusoidal path in the Radon plane. The matched filter for this is to integrate through the Radon plane of the entire ground patch. This is the matched filtering interpretation of image reconstruction given in [60]. Unfortunately, as elucidated by [18] in a different context, the SAR must operate in a multiple-target environment[12] and the sinusoidal traces from all the other reflectors in the scene overlap with the one being estimated, causing the $(\pi|r|)^{-1}$ spread. (The traces from two distinct reflectors will intersect twice over a look-angle variation of 2π.) To this extent, the interpretation of image reconstruction as matched filtering is faulty. The (one-dimensional) filtering operation of convolution-backprojection is thus seen to eliminate the effect of the overlapping traces. An important thing to notice here is that the matched filtering operation takes place across all pulse returns or range traces (to use SAR terminology) and so is unlike the

[12] Rihaczek [18] writes on the fact that the methods of detection (matched filtering) and the goals of resolution are often inconsistent with one another. Acknowledging the near-imperative of using matched-filter receivers, he writes, "For target resolution, it thus has been necessary to use a matched-filter receiver and optimize the waveform so as to reduce the mutual interference between targets, hopefully to the point where it becomes negligible." It is interesting to note that if one models radar data collection as a cascade of linear systems, that is, roughly, the transmitter (including antenna), the ground patch, and the receiver (including antenna and reconstruction algorithm, but not a modulus conversion of the data for display or post-processing, which is nonlinear), one can mathematically move the impulse response of the convolving filter used in the reconstruction, (2. 5), through to the transmitter. The result is that, for this application, the signal which eliminates interference between targets is this convolution kernel.

usual matched filtering done on a single return to generate a single range trace and which is based on maximizing a signal-to-noise ratio at one instant.

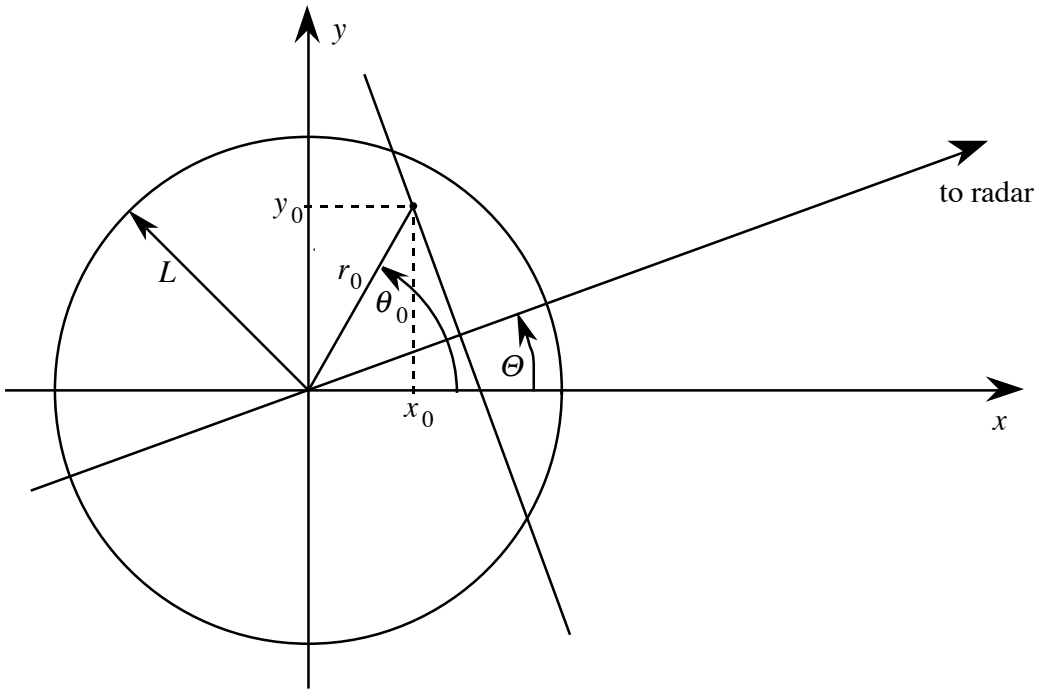
The matched filtering discussion of [18] is more traditional with respect to radar than that of [60], but more general than usual, as mentioned. The movement of the target is assumed to be slight enough that the complex envelope is not affected and the receiver implements a matched filter accordingly, that is, the range is estimated by autocorrelation with the transmitted signal, either by a filter or a bank of active correlators. In the analysis of [18], the output of this filter is then calculated on the basis of the influence on the RF carrier of the range derivatives. If it is necessary to know range derivatives, a multidimensional filter bank is required. For example, if active correlators are used to estimate delay, then the output of each correlator must be fed into a bank of filters to estimate range rate, or velocity. Likewise, the output of each velocity filter would be fed to a range acceleration filter, and so on. The complexity grows quickly if more than range and range rate are required to be known. The form of each of these filters is of the inner product type. The performance of the range derivative filters as maximum signal-to-noise ratio detectors is not given. The range of applicability of this theory is apparently only over a single pulse return, since it is restricted to situations which involve small time-bandwidth products, although this may be extendible.

It may be possible to combine the tomographic matched filter interpretation of [60] (many pulses, noncoherent) with the traditional but general matched filter of [18] (one pulse, coherent). In SAR, if the platform motion is known, then the range and range derivatives of any point in the ground patch are known exactly. Therefore, a prototype function for each resolution cell can be calculated across all returns and the reflectivity of that cell can be estimated as the result of the appropriate inner product computation. It should be possible to develop a procedure for a wide range of conditions involving wavefront curvature and antenna and propagation effects. Such algorithms should also automatically account for range walk, Doppler shifts, and Doppler spreading of wideband signals.

The previous paragraph will be elaborated somewhat; also, this is an opportune point to highlight one of the differences between the noncoherent aspect of X-ray tomography and the coherent aspect of SAR. (With the impulsive illumination such as is used throughout this dissertation, this difference tends to become obscured.) Two waveforms will be used; the first is a sinusoidal wave and the second is a pulsed sine wave, as a sufficient example of traditional range processing. Plane waves will be used, but extension to other situations is straightforward. First, let the transmitted signal be

$$s(t) = \exp(j\omega_0 t) .$$

In Fig. 6. 4, the two-way delay from a point target at (x_0, y_0), after removing the amount $2R/c$ which sets the reference to the patch center, is



$$u_0(t) = \exp\left\{ j\omega_0 \left[t + \frac{2r_0}{c} \cos(\Theta - \theta_0) \right] \right\}.$$

After imparting a constant rotation between the ground patch and the radar of $\Theta = \Omega t$ and downconversion, the baseband return signal is

$$u_{0b}(t) = \exp\left[j\omega_0 \frac{2r_0}{c} \cos(\Omega t - \theta_0) \right]. \tag{6. 3}$$

This will be found to be useful shortly in the pulsed-sinusoidal illumination. However, it is interesting to wonder whether this type of signal might be useful in itself. Consider attempting to reconstruct the ground patch reflectivity at the coordinates (x_0, y_0). One might consider computing the inner product

$$\int_0^{2\pi} u\left(\frac{\Theta}{\Omega}\right) \exp\left[-j\omega_0 \frac{2r_0}{c} \cos(\Theta - \theta_0) \right] d\Theta$$

where $u(t)$ is the summation of returns from the entire ground patch at baseband. The integration range is 2π to maintain consistency with earlier policy. An impulse response over r and θ could be measured by computing

$$\int_0^{2\pi} \exp\left[j\omega_0 \frac{2r}{c} \cos(\Theta - \theta) \right] \exp\left[-j\omega_0 \frac{2r_0}{c} \cos(\Theta - \theta_0) \right] d\Theta \, .$$

There appears to be a question concerning the resolving ability of such a method in the Θ variable, but this would be easy enough to ascertain. Also, the problem of mutual interference of multiple targets remains but might be solvable by methods discussed above. It is noted that a linearization of the sinusoidal frequency modulation of (6. 3) for small deviations in θ gives the familiar linear frequency modulation. This type of filtering is of the traditional one-dimensional type of [18] but with this formulation accommodates the full range of modulation caused by look-angle variation.

Next, a pulsed sine-wave illumination is considered. Let the transmitted signal be the single tone burst

$$s(t) = \text{rect}\left(\frac{t}{T} \right) \exp(j\omega_0 t)$$

where

$$\text{rect}(t) = \begin{cases} 1, & -\frac{1}{2} \le t \le \frac{1}{2} \\ 0, & \text{otherwise} \end{cases} \, .$$

Normally, $T \gg 2\pi/\omega_0$ meaning that there are many cycles of the RF signal contained with a pulse, and

$$T \ll \frac{1}{f_{D,\,max}} , \qquad f_{D,\,max} = \frac{4\pi L f_0 \Omega}{c} , \tag{6. 4}$$

the latter being an expression for the maximum Doppler frequency which can be found from (6. 3). The significance of the last condition will be seen shortly. The received baseband signal due to a point scatterer at (x_0, y_0), assuming negligible distortion to the envelope due to target motion over its duration, is

$$v_b(t) = \text{rect}\left[\frac{t + \frac{2r_0}{c} \cos(\Omega t - \theta_0)}{T} \right] \exp\left[j\omega_0 \frac{2r_0}{c} \cos(\Omega t - \theta_0) \right] \tag{6. 5}$$

which, with the condition (6. 4), is seen to be essentially a sample of the continuous signal (6. 3). The same result, that is, that (6. 5) is essentially a sampled version of (6. 4), still holds with range-compression techniques and although it glosses over most of range compression theory, it is adequate for the present purpose. Usual practice in SAR is to stack the range trace from each pulse by

forming a two-dimensional recording, instead of keeping the signal from all look angles in one dimension, as in (6. 5). Toward this end, let T_p be the pulse repetition period and let the pulses be indexed by an integer n. With the RF pulse train so defined, the periodically-sampled one-dimensional signal at RF is

$$v(t) = \sum_{n=0}^{N-1} \text{rect}\left[\frac{t - nT_p + \frac{2r_0}{c}\cos\left(\Omega nT_p - \theta_0\right)}{T}\right] \exp\left\{j\omega_0\left[t - nT_p + \frac{2r_0}{c}\cos\left(\Omega nT_p - \theta_0\right)\right]\right\}.$$

After converting to baseband, the two-dimensional signal is formed by placing pieces of $v(t)$ of length T_p next to one another, forming a variation indexed by n, as in

$$v_b(t, n) = \text{rect}\left[\frac{t + \frac{2r_0}{c}\cos\left(\Omega nT_p - \theta_0\right)}{T}\right] \exp\left[j\omega_0\frac{2r_0}{c}\cos\left(\Omega nT_p - \theta_0\right)\right]. \tag{6. 6}$$

The first term is a time-dependent narrow pulse the position of which varies sinusoidally with n. The second term is the sinusoidally-modulated Doppler [13] which depends on n but not t. (This is perhaps appropriately called interpulse Doppler.) This is the complex-valued projection of a point scatterer. By taking the liberties of connecting the discrete samples with line segments and making the pulse length so short that it does not appear as a separate entity, the above is plotted for a particular point scatterer (at some unspecified distance on the positive y-axis) in Fig. 6. 5. Comparison with the projection due to an impulse, as mentioned in earlier chapters, is invited. The essential difference, assuming a small (compressed) pulse length, is the right-hand term in the above, constituting a complex Doppler-induced modulation. It is easy to add the amplitude modulation that would be caused by antenna shading.

With the above development and the problem of overlap of multiple targets notwithstanding, the matched filtering operation for this style of recording can apparently be implemented by integrating along the sinusoidal trace with a weighting that is the complex conjugate of the right-hand side of (6. 6).

To summarize, there are what may be thought of as matched filtering ideas which have not been fully used in SAR image reconstruction, either as actual algorithm development aids or as aids to conceptual understanding.

6.3.4 Extended-support SAR

A method for potentially providing extended Fourier domain support for monostatic spotlight SAR under plane wave conditions and restricted look angle variation which capitalizes on intra-

[13] "Doppler" is used rather loosely here—as usual, a strict interpretation of Doppler requires that a time dependence be imposed by having the radar transmit pulses uniformly in time. This is sometimes a convenient artifice, since the important feature is the change in look angle Θ between pulse transmissions.

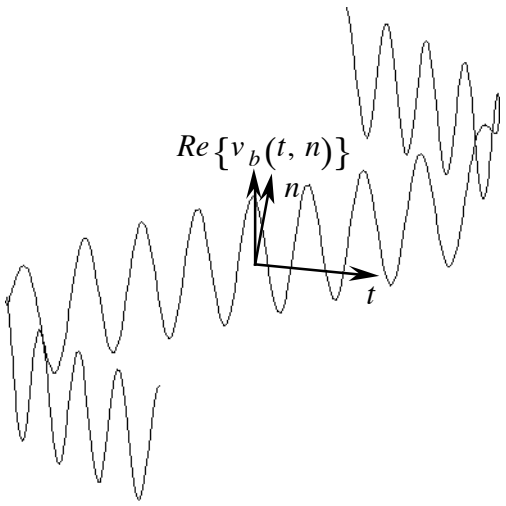


Fig. 6. 5. Perspective plot of the real part of the complex-valued return from a point reflector on the positive *y*-axis.

pulse Doppler will now be presented. (It also can be applied to other situations if spatial-domain reconstruction algorithms are employed. However, it is useful to be able to draw on Fourier transform concepts in this introduction.) The usual geometry with a rotated (x', y')-coordinate system is assumed. Also, assume that the radar's relative motion, at least for the time required to transmit and receive a signal, is $\Theta = \Omega t$, with Ω constant. With the radar at a distance sufficient to justify the plane wave condition, contours of equal Doppler shift across the ground patch, the so-called isodops, are straight lines parallel to the x' axis. This is the specialization for a very distant radar of the fact that isodops are families of hyperbolas—see [62], for example. Showing the specialized case is an easy exercise in basic vector mechanics. Therefore, each reflector will induce a Doppler shift on the reflected signal that is proportional to y', independent of x', and coded in complex amplitude by the reflector's reflectivity (and round-trip delay). This fact is made more useful if a waveform is selected to exploit this intrapulse effect; this waveform can be a sine wave modulated by a relatively long gate. The signal returned is a sum of all such Doppler-shifted signals. Fourier transformation of each range trace results in the projection of the ground patch along lines at an angle $\Theta_i = \Omega t_i$ where t_i is the nominal time of pulse transmission and reception. This is in contrast to ground patch projections along lines at an angle $\Theta_i + \pi/2$ which result from the usual pulse compression techniques. It therefore appears that the Fourier domain support available for inversion can be extended to include the shaded regions in Fig. 6. 6, shown for a restricted range of look angles which vary around $\Theta = 0$. Whereas the angular displacement of the two regions will always be $\pi/2$, the radial extents can differ. As usual, Doppler processing benefits in terms of resolution from a longer data record. This might become a problem for the current scheme, in extreme situations, due to projection lines effectively rotating during the dwell time of such a long pulse, resulting in angular smearing over wedge-shaped sections centered on the ground patch. This might be

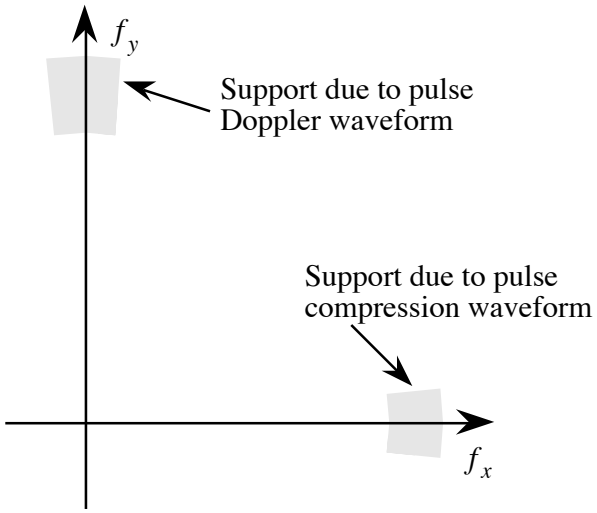


Fig. 6. 6. An example of Fourier domain support from using both pulse compression and pulse Doppler waveforms for a SAR viewing with restricted look angle in the nominal direction $\Theta = 0$. The frequency variables are f_x and f_y, corresponding to the image coordinates x and y.

removed by a circular deconvolution in a manner reminiscent of correction for finite beam and detector widths in X-ray tomography. Also, non-Fourier spectrum estimation techniques might be used to advantage on shorter data records.

As an example of the above, consider a point target at some unspecified distance on the positive y-axis. Even for a long pulse (in the Doppler resolution sense— $T \approx 1/\Delta f_D$ where Δf_D is the desired Doppler resolution), the pulse will probably be short enough that the Doppler shift is approximately constant within that pulse. With this, (6. 3) can be adapted by approximating its Doppler shift as a piecewise constant function. Differentiating the phase part of (6. 3) with respect to time gives the instantaneous radian Doppler frequency as

$$\omega_D = 2\pi f_D = -\frac{2\Omega r_0}{c}\sin(\Theta_i - \theta_0)$$

so that the two-dimensional (stacked) signal can be represented by

$$u_b(t, n) = \text{rect}\left[\frac{t + \frac{2r_0}{c}\cos\left(\Omega n T_p - \theta_0\right)}{T}\right]\exp\left[-\frac{2\Omega r_0}{c}\sin\left(\Omega n T_p - \theta_0\right)t\right].$$

The functions

$$\binom{\cos}{\sin}\left[-\frac{2\Omega r_0}{c}\sin\left(\Omega n T_p - \theta_0\right)t\right]$$

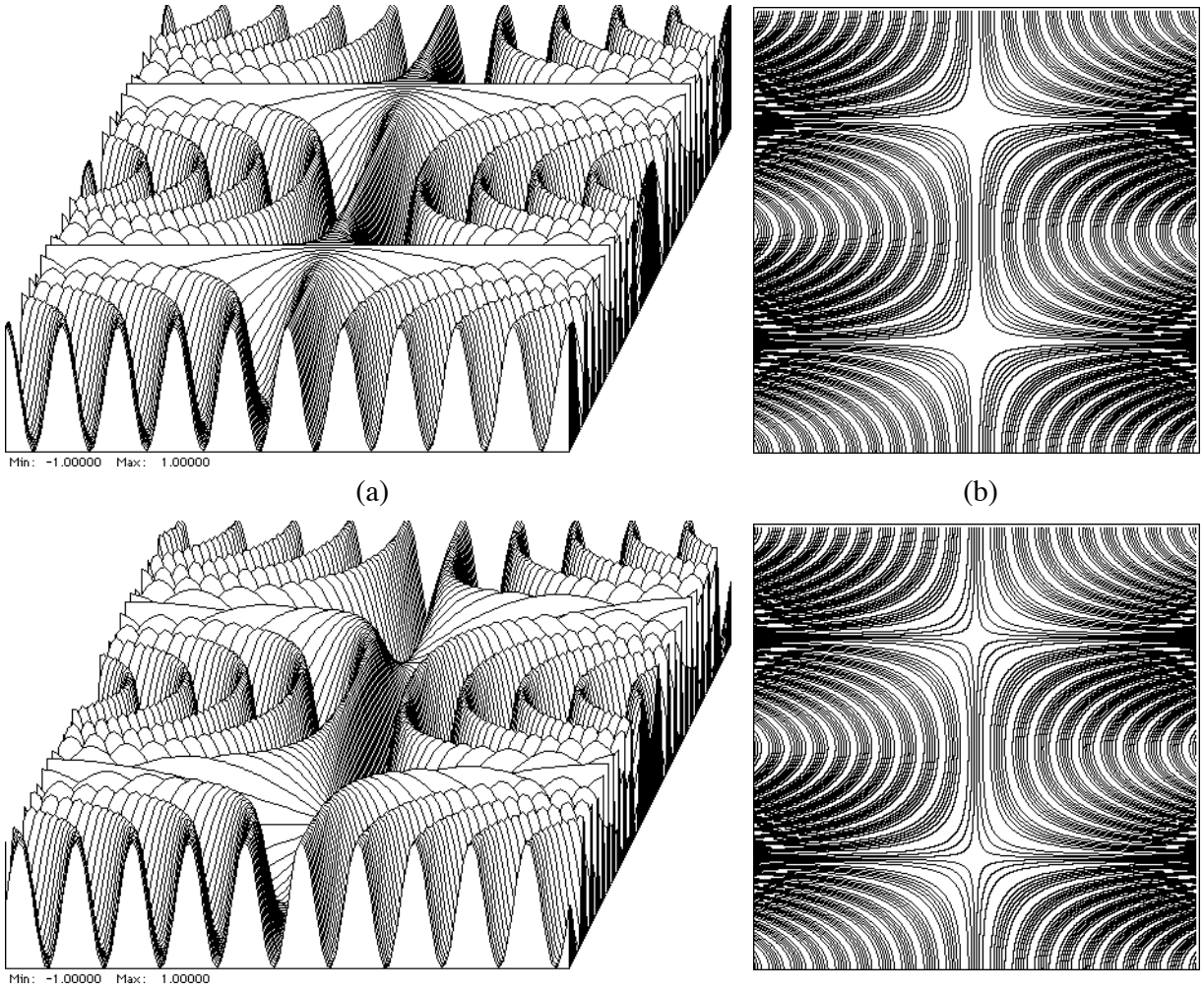


Fig. 6. 7. Example of Doppler-shifted range traces of a point target from a constantly-moving SAR. Horizontal variable is "fast time" and depth variable is "slow time." See the text for ranges of these variables. (a) Perspective plot of the real part of the complex return. (b) Contour plot of (a). (c) Perspective plot of the imaginary part of the complex return. (d) Contour plot of (c).

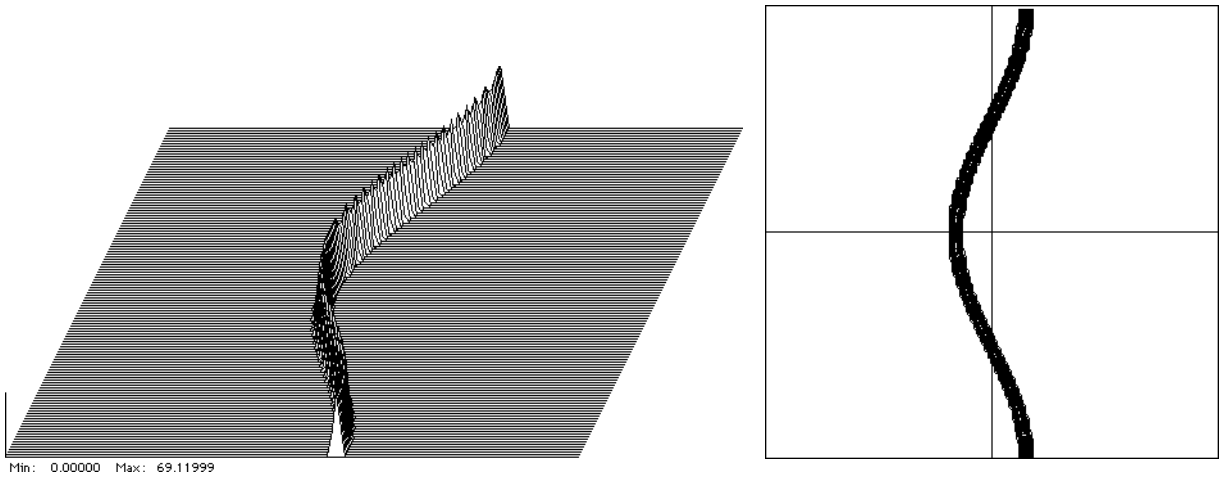
are plotted in Fig. 6. 7 to illustrate the real and imaginary parts of this Doppler signal. The horizontal variable and its range are

$$-10\pi \le \frac{2\Omega r_0 t}{c} \le 10\pi$$

and the depth variable and its range are

$$0 \le \Omega n T_p \le 2\pi$$

with $\theta_0 = \pi/2$. (The range in $\Omega n T_p$ used is inconsistent with the earlier assumption but is useful here for illustrative purposes.) Using Fig. 6. 7 (a) as the real part and Fig. 6. 7 (c) as the imaginary



(a) (b)

Fig. 6. 8. DFT along the horizontal direction of each range trace of Fig. 6. 7 using Fig. 6. 7 (a) as the real part and Fig. 6. 7 (b) as the imaginary part of the input to the DFT. This shows the quadrature relationship of the sinusoidal trace obtained using a modified pulse Doppler technique for SAR imaging, relative to that obtained using conventional pulse compression techniques. (a) Perspective plot. (b) Contour plot.

part, Fig. 6. 8 shows the result of computing the DFT (with Hamming window) for each range trace, that is, along the horizontal variable in Fig. 6. 7. This shows the predicted result. Notice that this sinusoidal trace stands in quadrature with respect to the trace that would result from a pulse compression waveform from the same target when measured over the same range of look angle.

A few final comments will be made. This kind of imaging, if practical, would experience a coherent interference phenomenon similar to imaging with a pulse compression system, only here the interference is along isodops, not cross-range. Both phenomena take place on the wavelength scale, but the scale for the present system is half because of the two-way delay. (Of course, it is envisioned that both types of waveforms could be used together.) Perhaps sinusoids of different frequencies could be transmitted, using the common technique of frequency diversity to reduce speckle. Here, the increased bandwidth should come at low additional cost because the hardware is already in place to process the pulse compression signal. It is fortuitous that the additional support available is at 90° from the usual support. Although a resolution analysis is not done here, it seems, at least in the absence of speckle, that the 90° displacement is optimal for resolution. This stands to reason on the basis of backprojection being a kind of triangulation technique. Interestingly, the range attenuation problem with the Doppler projections becomes analogous to the antenna shading problem with range projections, and the antenna shading problem apparently becomes as trivial to correct with Doppler projections as the range attenuation problem is with range projections. Finally, system considerations for a specific application would decide the practicality of this combined Doppler-pulse compression method.

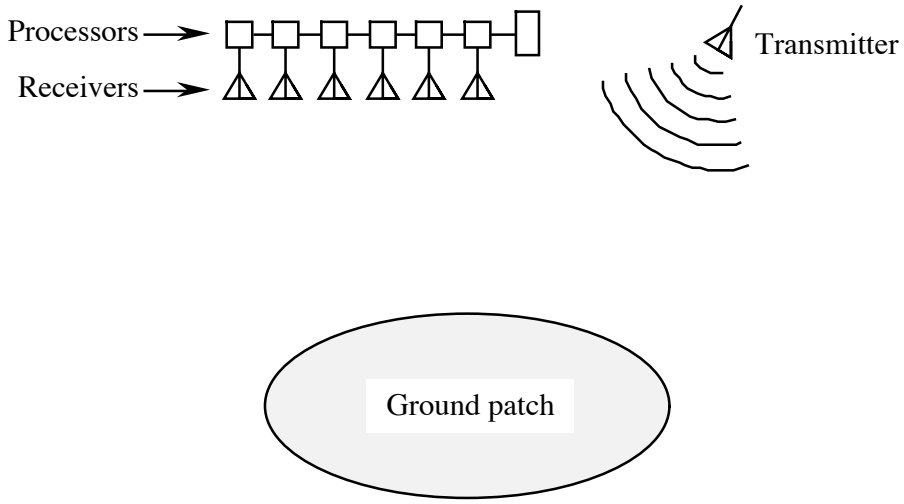


Fig. 6. 9. Concept for an array of receiving antennas.

6.3.5 Towed arrays

The next imaging concept includes the use of a "towed array" (to borrow sonar terminology) of receiving antennas as in Fig. 6. 9. A tomographic interpretation of this scenario is helpful but a Doppler interpretation is more of a hindrance. Consider, for example, a low-Earth orbiting SAR with a linear velocity of 17,600 mi/h and a PRF of 1500 pps (approximately the SEASAT PRF). The distance traveled between pulse transmissions is about 17 ft. The same angular sampling increment could be obtained from an array of receiving antennas at double that spacing, 34 ft. The doubling in spacing is due to the fact that the receiving antennas are essentially multistatic and the sampling angle is half of each bistatic angle. Such an array would be useful in LPI situations where the PRF could be lowered by several integer factors. In principle, with an array long enough, data for an entire image could be collected from the reflections from only one pulse. Other advantages include decreased power requirements for the transmitter and a significantly improved situation with respect to multiple-time-around echoes. In an airborne application, the array might actually be towed, either behind the transmitter or from another platform; in space, naturally, actual towing would not be necessary. The prospects for distributed processing are interesting. As suggested in Fig. 6. 9, each receiver could have its own processor which could filter its own projection and compute the backprojection. A separate processor would add the backprojections together. Such an arrangement might also affect presumming and other operations.

6.3.6 Transformation of projections

Another image reconstruction scheme is to convert the available projections, for example, circular-arc projections, into the straight-line projections that correspond to the image or ground patch, and then to reconstruct using ordinary convolution-backprojection. It is conjectured that the

transformation is itself an integral over a two-parameter set of curves. The method, if viable, does not appear to have any computational advantage over methods developed in this dissertation, but could provide an alternative under conditions that are otherwise difficult, and may have use as a conceptual tool. It may prove to be equivalent to the algorithm mentioned at the end of Section 6.1 which would decompose a spherical wave (for example) into several plane waves.

6.4 Extension of Current Work

This section will list a set of more or less obvious extensions of the work presented in this dissertation (not including the previous section). Although items on this list may be obvious, this is not to imply that they would be easy to do.

The problem of autofocusing (removal of residual motion errors based on collected data, after inertial data have been used) [2] has not been addressed. This problem may take on a new form when cast in the context of tomographic-style image reconstruction.

Presumming [25] also may acquire a new character in the tomographic setting. Normally, presumming is used to effectively narrow the antenna beamwidth to reduce the amount of data which is collected using a PRF set according to adequate Doppler sampling when considering the real antenna beam spread. Tomographically, presumming might involve averaging (or otherwise combining) several projections, but this would cause the same kind of smearing over a range of angles that was mentioned earlier in regard to Doppler-imaging for extended Fourier support when using unusually long pulses from a fast-moving platform. This circular blurring could presumably be removed as well.

The present work has assumed a flat Earth and a radar at zero elevation. While this is convenient, ultimately the curvature of the Earth may have to be taken into account. The projections from a monostatic SAR could then be modeled as resulting from spherical waves intersecting a spherical Earth, so that they are still circular arcs. The equivalent result for bistatic SAR is not as easy to discern. If available, local topographic data could be included in the reconstruction, but perhaps at the cost of greatly-increased complexity. Towards this end, [61] discusses the use of two receiving antennas and an interferometric technique to measure topographic information along with SAR information.

A conceptually straightforward extension of this or any other work on spotlight SAR is to simultaneously image multiple ground patches using multiple beams of an electronically-scanned array antenna. A schedule for transmitting and receiving would have to be established to avoid interference between returns from various places. One scheme would be to map a continuous strip with spotlight-mode resolution by forming a mosaic of spotlight images. A more elegant way of combining data to form the strip image is probably available, however. The data processing requirements for such a radar would be heavy but easily partitioned, at least if the mosaic route were taken.

It may at times be necessary to account for other types of attenuation besides spherical spreading. Atmospheric attenuation is one possibility, depending on the radiated frequencies involved. What would amount to another kind of range-dependent attenuation would result from the elevation antenna pattern illuminating the ground patch nonuniformly from an elevated position. Normally, antennas with a cosecant-squared elevation pattern are used for this correction, but doing the correction as part of the reconstruction algorithm is an alternative.

Many of the algorithms that have been developed or proposed here require more computation than most Fourier transform-based algorithms due mostly to the lack of a "fast" algorithm. Partly because of this and partly because of the increased coverage and resolution of some applications, it is becoming increasingly desirable to map image reconstruction algorithms to systolic arrays or other appropriate computer architectures. The convolution-backprojection style of algorithms has features which make partitioning of this kind practical, as discussed in [32].

APPENDIX A

ON THE APPROXIMATE EQUIVALENCE OF MATCHED FILTER AND STRETCH RECEIVERS FOR LINEAR FREQUENCY MODULATED SIGNALS

TWO popular types of radar receiver, the matched filter receiver and the stretch receiver, are compared in the absence of noise and are shown to be approximately equivalent as long as the time-bandwidth product of the transmitted signal is large and the variation in the delays of the reflected signals is small compared to the duration of the transmitted signal.

A.1 Introduction

Signals having linear frequency modulation (LFM) are favored in certain types of radars due to their special properties [18] and ease of generation [20] compared to some other signals. Systems using LFM signals are in widespread use and are well understood; as a result, several methods are used for processing reflected signals in such radar receivers.

Perhaps the two most common receivers for LFM pulsed radars are the matched filter receiver and the so-called stretch receiver. The former is an implementation of a matched filter [91] for the transmitted waveform [20], and as such is based on statistical correlation functions. The latter is credited by Wehner [20] to Caputi [92], although a version of this receiver is presented in [93]. Also, a version of the stretch receiver is discussed in [58]. Both receivers are an implementation of the pulse compression technique.

Historical aspects aside, some radar engineers have noticed that these two types of receiver at times give similar outputs. This paper confirms those observations via a straightforward derivation. In particular, conditions under which the approximate equivalence holds are clearly stated, so that departures from approximate equivalence may be predicted. The analysis does not include the effects of noise or equipment limitations such as the effects of slight nonlinearities in the frequency modulation, nor are the receivers compared at a hardware level.

The following signals, some being complex-valued or having one-sided spectra, will be used freely throughout. The transmitted signal is

$$s(t) = u(t) \exp\left(j2\pi f_0 t\right)$$

where f_0 is the carrier frequency and the modulation function is

$$u(t) = a(t)\exp\left(j\pi k t^2\right) \tag{A. 1}$$

where $a(t)$ is the pulse shape and k is the chirp rate or sweep rate in Hz/s. In Section A.2, $a(t)$ is taken as rect(t/T), where T is the pulse duration and

$$\text{rect}(t) = \begin{cases} 1, & |t| \le \frac{1}{2} \\ 0, & \text{otherwise} \end{cases} .$$

Two signals which are reflected from point scatterers will be considered. In both, the effects of propagation attenuation, antenna weighting, and reflection coefficient will be ignored, as these may all be lumped into a single complex constant (per scatterer), and thus have no bearing on the present problem. The first reflected signal is actually a fictitious signal, one that *would* be reflected from a reference range if there were a point scatterer there. This reference signal is

$$\begin{aligned} r_0(t) &= s(t - T_0) \\ &= u_0(t)\exp\left(j2\pi f_0 t\right) \end{aligned} \tag{A. 2}$$

where

$$u_0(t) = a(t - T_0)\exp\left(-j2\pi f_0 T_0\right)\exp\left[j\pi k(t - T_0)^2\right] .$$

Here, T_0 is the two-way signal delay of the reference signal from the time of transmission. Similarly, the signal which is reflected from a point scatterer at delay T_1 is

$$\begin{aligned} r_1(t) &= s(t - T_1) \\ &= u_1(t)\exp\left(j2\pi f_0 t\right) \end{aligned}$$

where

$$u_1(t) = a(t - T_1)\exp\left(-j2\pi f_0 T_1\right)\exp\left[j\pi k(t - T_1)^2\right] .$$

Further, as a convention, upper-case symbols will represent the Fourier transforms of their lower-case counterparts.

A.2 Rectangular Pulses

This section derives the results for a transmitted rectangular pulse.

A.2.1 Matched filter receiver

The matched filter receiver is modeled here as simply a matched filter for the reflection from the fictitious scatterer at the reference range. With $a(t) = \text{rect}(t/T)$, then according to the matched filter principle for delayed signals (see, e.g ., [16]), in order for the filter output to be maximized at time $t = 0$, its impulse response should be

$$h(t) = r_0^*(-t)$$

$$= \text{rect}\left(\frac{-t - T_0}{T}\right) \exp\left[-j2\pi f_0(-t - T_0)\right] \exp\left[-j\pi k(-t - T_0)^2\right]$$

$$= u_0^*(-t) \exp(j2\pi f_0 t) \qquad \text{(A. 3)}$$

which is seen to be a time-reversed, complex-conjugated version of the baseband reference function, modulated upwards in frequency by f_0. The matched filter output is then

$$y_{mf}(t) = r_1(t) * h(t)$$

or

$$Y_{mf}(f) = R_1(f)\, U_0^*(f - f_0)$$

where * denotes convolution and * denotes complex conjugation. However, since this signal is at the carrier frequency and it is normally desired to use the received signal at baseband, it is desirable to depart from the strict interpretation of the matched filter concept and allow for downconversion in frequency by f_0. This gives the baseband signal

$$z_{mf}(t) = \left[\exp(-j2\pi f_0 t)\, r_1(t)\right] * u_0^*(-t)$$

or

$$Z_{mf}(f) = R_1(f + f_0)\, U_0^*(f)$$

which implies the block diagram of Fig. A. 1 for this receiver. An alternate form of the matched filter receiver output is

$$z_{mf}(t) = u_1(t) * u_0^*(-t)$$

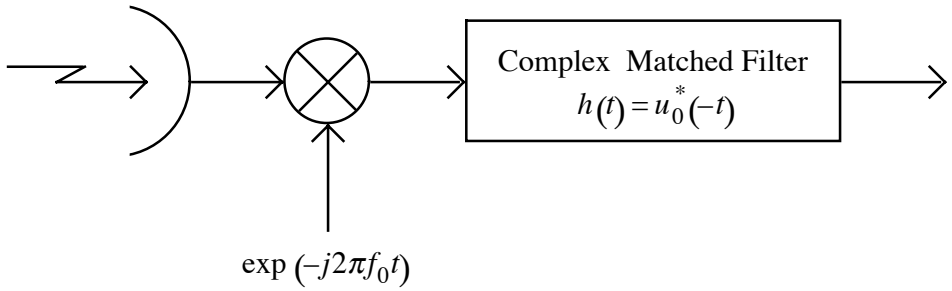


Fig. A. 1. Block diagram of a matched filter receiver for linear frequency modulation signals.

which is seen to be the output of a filter matched to the modulation function of the transmitted signal.

To derive the details of $z_{mf}(t)$, it is necessary to know the Fourier transform of $u(t)$, since $u_0(t) = u(t - T_0) \exp(-j2\pi f_0 T_0)$ and $u_1(t) = u(t - T_1) \exp(-j2\pi f_0 T_1)$. Unfortunately, this involves Fresnel integrals, which must be computed numerically. However, for the case of large time-bandwidth product (TB, where $B = kT$ is the swept bandwidth) signals, there is a convenient result. In general, for large TB, the signal

$$m(t) = a(t) \exp\left(j\pi k t\right)^2$$

has a Fourier transform [18]

$$M(f) = \frac{1}{\sqrt{|k|}} a\left(\frac{f}{k}\right) \exp\left(-j\pi \frac{f^2}{k} \pm j\frac{\pi}{4}\right) \tag{A. 4}$$

where the phase factor $\pi/4$ takes the same sign as k. With $a(t) = \text{rect}(t/T)$, the result

$$U(f) = \frac{1}{\sqrt{|k|}} \text{rect}\left(\frac{f}{kT}\right) \exp\left(-j\pi \frac{f^2}{k} \pm j\frac{\pi}{4}\right)$$

is obtained, whereby

$$Z_{mf}(f) = U_1(f)\, U_0^*(f)$$

$$= \exp\left[j2\pi f_0 (T_0 - T_1)\right] \exp\left[j2\pi f (T_0 - T_1)\right] \frac{1}{|k|} \text{rect}\left(\frac{f}{kT}\right). \tag{A. 5}$$

Inverse transforming (A. 5), the time-domain output of the matched filter receiver is found to be

$$z_{mf}(t) = \exp\left[j2\pi f_0 (T_0 - T_1)\right] T \frac{\sin\left[\pi k T (t + T_0 - T_1)\right]}{\pi k T (t + T_0 - T_1)}. \tag{A. 6}$$

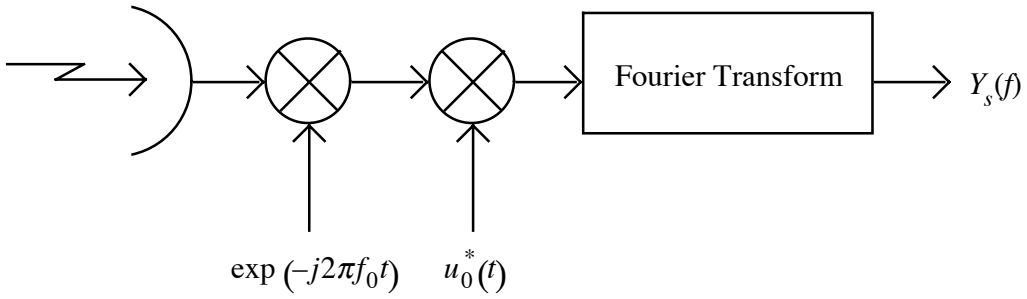


Fig. A. 2. Block diagram of a receiver for linear frequency modulation signals.

This signal achieves its maximum amplitude when $t = T_1 - T_0$, as desired.

A.2.2 Stretch receiver

The stretch receiver is specified by stating that the reflected signal is mixed (multiplied) by the complex conjugate of a reference function, the same reference function $r_0(t)$ as in (A. 2), and computing the Fourier transform of the product. The resulting signal, reinterpreted as a time-domain signal if desired, is the receiver output. Examining the conjugate of $r_0(t)$, it is seen that the mixing can be done in two stages. The first stage is simply a downconversion to baseband. The second stage is a multiplication by the conjugate of the reference modulation function $u_0(t)$. The block diagram of the stretch receiver is shown in Fig. A. 2.

The product signal is

$$y_s(t) = u_1(t)\, u_0^*(t)$$

$$= \text{rect}\left(\frac{t-T_0}{T}\right)\text{rect}\left(\frac{t-T_1}{T}\right)\exp\left[j2\pi f_0 (T_0 - T_1)\right]\exp\left\{j\pi k\left[(t-T_1)^2 - (t-T_0)^2\right]\right\}. \quad \text{(A. 7)}$$

The product of the rect(·) functions can be found to be

$$\text{rect}\left(\frac{t-T_0}{T}\right)\text{rect}\left(\frac{t-T_1}{T}\right) = \begin{cases} \text{rect}\left[\dfrac{t-(T_0+T_1)/2}{T-|T_0-T_1|}\right], & |T_0 - T_1| < T \\ 0, & \text{otherwise} \end{cases}$$

that is, nominally another rect(·) function with a leading edge at $t_1 = (T_0 + T_1 + |T_0 - T_1| - T)/2$ and a trailing edge at $t_2 = (T_0 + T_1 - |T_0 - T_1| + T)/2$. When there is no overlap, the output is zero. Using t_1 and t_2 as integration limits, the Fourier transform of (A. 7) is found to be

$$Y_s(f) = \begin{cases} \exp\left[j2\pi f_0 (T_0 - T_1)\right] \exp\left[-j2\pi f (T_0 - T_1)/2\right] (T - |T_0 - T_1|) \\ \qquad \bullet \dfrac{\sin\left\{\pi\left[k(T_0 - T_1) - f\right] (T - |T_0 - T_1|)\right\}}{\pi\left[k(T_0 - T_1) - f\right] (T - |T_0 - T_1|)} \quad , \ |T_0 - T_1| < T \quad . \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad , \text{ otherwise} \end{cases} \tag{A. 8}$$

The factor $(T - |x|)$, with $x = T_0 - T_1$, is a triangular window in x which accounts for the expected overlap loss of signals which are returned from points away from the reference point.

A.2.3 Approximate equivalence

One form of the approximate equivalence of these two receivers can be seen by comparing the magnitudes of (A. 6) and (A. 8) under the condition $|T_0 - T_1| << T$, which restricts (A. 8). With this restriction,

$$|Y_s(f)| \approx \left| T \frac{\sin\left[\pi k T(-f/k + T_0 - T_1)\right]}{\pi k T(-f/k + T_0 - T_1)} \right| .$$

and

$$\left| z_{mf}\left(-\frac{f}{k}\right) \right| \approx |Y_s(f)| .$$

The relationship between the two receivers is actually somewhat stronger than this, since (A. 6) and (A. 8) differ, with $|T_0 - T_1| << T$, only by the second exponential of (A. 8). This exponential term can be removed in some cases, making the two receivers approximately equivalent even in their complex outputs. If the condition $|T_0 - T_1| << T$ does not exist, the overall $\sin(x)/x$ nature of (A. 8) persists, only it is scaled in amplitude by the triangular factor $(T - |T_0 - T_1|)/T$ and in its independent variable by the inverse of the same factor.

A.3 Extension to Non-Rectangular Pulses

The above results regarding approximate equivalence are essentially extendible to LFM waveforms with arbitrary envelope functions under nearly the same conditions. Since rectangular pulses result in receiver outputs which are proportional to a $\sin(x)/x$ form, other envelope shapes are sometimes used to decrease the sidelobe levels. Standard windowing theory applies in these cases. Commonly, the envelope has a relatively slowly varying time dependence and resembles one cycle of a raised cosine pattern, e g ., a Hamming form. The derivation is essentially the same as in Section A.2, except that $a(t)$ in (A. 1) is left unspecified. While it is not possible to derive all of the details of either the matched filter or the stretch receiver without a specific form for $a(t)$, certain

aspects become more obvious in the generalization.

A.3.1 Matched filter receiver

The more general result for the matched filter receiver is obtained, with $u(t) = a(t) \exp(j\pi kt^2)$, by using (A. 5), leading to

$$Z_{mf}(f) = \exp[j2\pi f_0(T_0 - T_1)] \exp[j2\pi f(T_0 - T_1)] |U(f)|^2 .$$

Using the large *TB* approximation (A. 4),

$$|U(f)|^2 = \frac{1}{|k|} \left| a\left(\frac{f}{k}\right) \right|^2$$

and so

$$Z_{mf}(f) = \exp[j2\pi f_0(T_0 - T_1)] \exp[j2\pi f(T_0 - T_1)] \frac{1}{|k|} \left| a\left(\frac{f}{k}\right) \right|^2 . \qquad \text{(A. 9)}$$

The time-domain signal is found to be

$$z_{mf}(t) = \exp[j2\pi f_0(T_0 - T_1)] \frac{1}{|k|} \mathsf{F}^{-1} \left\{ a\left(\frac{f}{k}\right) a^*\left(\frac{f}{k}\right) \right\}_{t \leftarrow t + T_0 - T_1} \qquad \text{(A. 10)}$$

where the notation on the right end means "replace t with $t + T_0 - T_1$ in the results of the Fourier transform." Using the following device, (A. 10) will be modified in anticipation of the results for the stretch receiver, to make the two results look as much alike as possible. For any Fourier- transformable function $b(f)$,

$$\mathsf{F}^{-1}\{b(f)\} = \mathsf{F}\{b(-f)\}$$

where the transform on both sides is understood to be with respect to f. Applied to (A. 10), this gives

$$z_{mf}(t) = \exp[j2\pi f_0(T_0 - T_1)] \frac{1}{|k|} \mathsf{F} \left\{ a\left(-\frac{f}{k}\right) a^*\left(-\frac{f}{k}\right) \right\}_{t \leftarrow t + T_0 - T_1} . \qquad \text{(A. 11)}$$

Again, the transform in (A. 11) is with respect to f.

A.3.2 Stretch receiver

The derivation for the stretch receiver begins with (A. 7). This yields

$$y_s(t) = a(t-T_1)\,a^*(t-T_0)\exp\left[j2\pi f_0(T_0-T_1)\right]\exp\left[j\pi k\left(T_1^2-T_0^2\right)\right]\exp\left[j2\pi k(T_0-T_1)\right]t \tag{A. 12}$$

which already bears some resemblance to (A. 9). Fourier transformation by the receiver results in the final output signal

$$Y_s(f) = \exp\left[j2\pi f_0(T_0-T_1)\right]\exp\left[-j\pi f(T_0+T_1)\right]\mathsf{F}\left\{a\left(t-\frac{T_1-T_0}{2}\right)a^*\left(t+\frac{T_1-T_0}{2}\right)\right\}_{f\leftarrow f-k(T_0-T_1)}. \tag{A. 13}$$

This equation contains the effects of no approximations (except for those inherent in the receiver model). Using (A. 13), the rectangular pulse form (A. 8) can be found in a straightforward manner, as can (A. 6) using (A. 11).

A.3.3 Approximate equivalence

A comparison of (A. 11) and (A. 13) shows that the two are at least similar. However, it seems useful to carry the results a little further in order to establish nearly the same level of approximate equivalence as was done in Section A.2. To facilitate this, the signal

$$\mathsf{F}\left\{a(t)\,a^*(t)\right\} = \mathsf{A}(f) \tag{A. 14}$$

is introduced, so that

$$\mathsf{F}\left\{a\left(-\frac{t}{k}\right)a^*\left(-\frac{t}{k}\right)\right\} = |k|\,\mathsf{A}(-kf)\,. \tag{A. 15}$$

By exchanging the roles of t and f in (A. 15) and applying it to (A. 11), the final result for the matched filter receiver is obtained as

$$z_{mf}(t) = \exp\left[j2\pi f_0(T_0-T_1)\right]\mathsf{A}\left[-kt-k(T_0-T_1)\right] \qquad (\text{large } TB)\,. \tag{A. 16}$$

Before applying (A. 14) to (A. 13), the assumption $|T_1 - T_0| << T$ needs to be made once again. Actually, the limits of this assumption would have to be checked for each $a(t)$: how large can $|T_1 - T_0|$ be before the relation

$$\mathsf{F}\left\{a\left(t-\frac{T_1-T_0}{2}\right)a^*\left(t+\frac{T_1-T_0}{2}\right)\right\}_{f\leftarrow f-k(T_0-T_1)} \approx \mathsf{F}\left\{a(t)\,a^*(t)\right\}_{f\leftarrow f-k(T_0-T_1)}$$

$$= \mathsf{A}\left[f-k(T_0-T_1)\right] \tag{A. 17}$$

becomes too poor. Normally, $a(t)$ varies slowly over its duration T, and any slight offset between two copies of $a(t)$ will affect the duration of the product more than the overall shape of the product. In fact, in the rectangular case of Section A.2, the overall shape of the product pulse was unchanged, since it too was a rectangular pulse. So as long as (A. 17) is satisfied, using (A. 14) in (A. 13) then results in the stretch receiver output

$$Y_s(f) = \exp\left[j2\pi f_0(T_0 - T_1)\right] \exp\left[-j\pi f(T_0 + T_1)\right] \mathsf{A}\left[f - k(T_0 - T_1)\right] \qquad (\text{small } |T_0 - T_1|) \,. \tag{A. 18}$$

The remaining differences between (A. 16) and (A. 18) are now briefly discussed. In the argument of the A function of (A. 16), the term $-kt$ indicates an axis reversal and scaling. The other term is the required signal delay which differentiates scatterers at different ranges. The scaling and axis reversal are considered nonessential in establishing approximate equivalence. In (A. 18), the surplus phase term was discussed in Section A.2. In the A term, there is no scaling or reversal of the independent variable. If only the magnitudes are compared, the matched filter output can be obtained from the stretch output simply by the substitution $f = -kt$.

A.4 Discussion

The results of Sections A.2 and A.3 show that, with the assumptions that the transmitted signal has a large time-bandwidth product and that the difference in signal delay between reflections from the fictitious reference scatterer and the scatterer of interest is small compared to the transmitted (uncompressed) pulse duration, then the matched filter receiver and the stretch receiver produce approximately equivalent outputs. Of course, linearity of the receivers allows the extension of the results to multiple scatterers or extended reflecting objects.

That the two receivers are similar can be appreciated in part by the following intuitive argument. Consider a vector space of finite-energy signals with an inner product

$$\langle v, w \rangle = \int_{-\infty}^{\infty} v(t)\, w^*(t)\, dt \,.$$

In general, a matched filter receiver computes the inner product of the received signal $r_1(t)$ with stored copies of all of the possible received signals, $x_i(t)$, $i = 1, 2, \ldots$, and selects the one which generated the largest inner product as that most resembling $r_1(t)$. The inner product may be computed by a bank of correlators or a bank of filters whose impulse responses are the time-reversed versions of the $x_i(t)$. In the special case in which all of the possible received signals are scaled and delayed versions of the transmitted signal, the bank of filters can be collapsed into a single filter, as in (A. 3). (While the matched filter has optimum detection properties in the face of additive white Gaussian noise, it does not necessarily have optimum resolution properties.) The stretch receiver,

on the other hand, first converts the range information T_1 (actually, $T_1 - T_0$, since T_0 is presumably known) into proportional frequency information by the dechirping stage (A. 7), as seen in the time-varying exponent in (A. 12), for example. For long signals of unknown frequency, the matched filter is one which computes the inner product with an ensemble of sinusoids,

$$\left\langle r_1, e^{j2\pi ft} \right\rangle = \int_{-\infty}^{\infty} r_1(t)\, e^{-j2\pi ft}\, dt$$

which validates the Fourier transform stage. Requiring that the transmitted pulse be long compared to the range of offsets of reflected signal delays assures that the Fourier analysis gives results that are nearly the same as if the pulse were infinitely long, as well as assuring that "edge effects" in the regions of no overlap with the reference function are sufficiently small.

Finally, it should be noted that these results are unique to LFM signals. This can be seen from consideration of the above paragraph with respect to the stretch receiver. Specifically, (A. 7), implemented for other types of modulation, would not proportionally translate range into frequency and therefore the Fourier transform stage would not properly decode the result.

APPENDIX B

FAR-FIELD CONSIDERATIONS FOR RADAR RECEPTION

THE PURPOSE of this appendix is to show that under ordinary operating conditions, a ground-illuminating radar is in the near field[14] of the reflected signal at least as long as the reflecting ground patch is in the far field of the transmitting antenna. This surprising result is not thought to be widely known.

A nominal geometry for a monostatic radar is shown in Fig. B. 1. The object being measured by the radar, here labeled "Ground Patch," is a distance R from the phase center of the radar antenna, which has an effective radiating aperture size of d linear units.

The ground patch is assumed to be in the far field of the transmitting antenna. In order for this to hold, all three of the following conditions must hold [94]:

$$R \gg d$$

$$R \gg \lambda$$

$$R > \frac{2d^2}{\lambda}.$$

Another set of conditions, more useful in practice, is

$$R > 5d$$

$$R > 1.6\,\lambda$$

$$R > \frac{2d^2}{\lambda}.$$

It is almost always the third of these conditions which is the most difficult to meet, so this will be used hereafter.

If the far-field conditions during transmission are met, then the antenna has a far-field beamwidth θ_r given by $\sin(\theta_r/2) = \lambda/d$. If $\lambda \ll d$, i.e., the beamwidth is rather narrow, then the beamwidth is approximated by $\theta_r = 2\lambda/d$. This causes an area on the ground of extent $D = R\theta_r = 2R\lambda/d$

[14] The use of the term "near field" here is meant to designate any part of the field that is not in the far field. This may be in violation of the strict meaning used in electromagmetics texts, but seems convenient here.

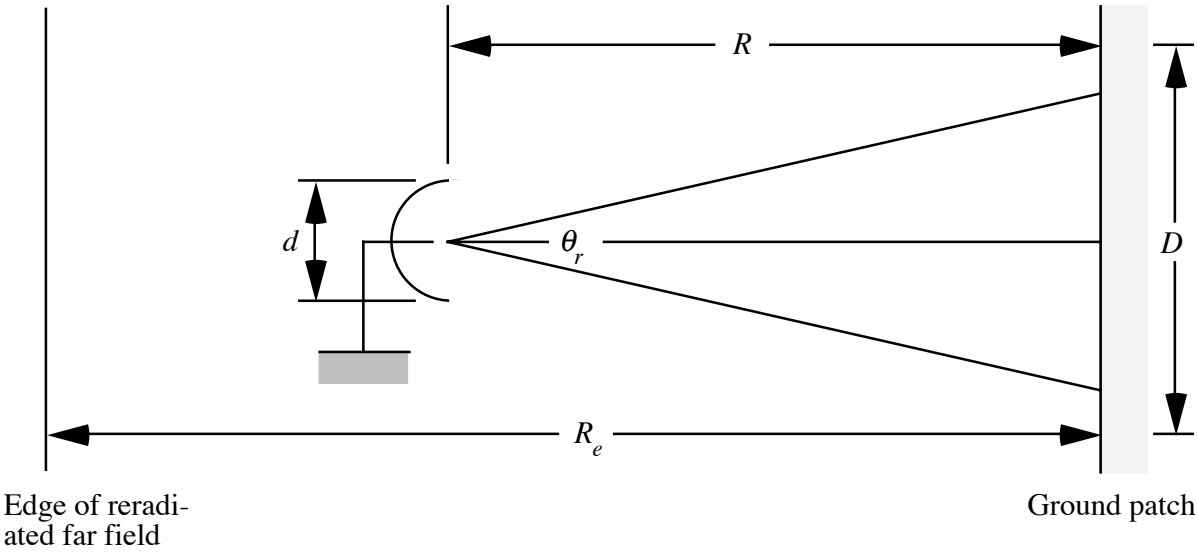


Fig. B. 1. Geometry showing the radar in the near field of the reradiated signal.

to be illuminated. Upon reradiation, this portion of the ground behaves like an antenna with an aperture of size D. Calling the edge of the far field of this reradiated energy R_e, the result $R_e = 2D^2/\lambda$ is obtained, or

$$R_e = \frac{8R^2\lambda}{d^2}.$$

This result shows that as the distance between the radar and the ground patch increases, the edge of the reradiated far field recedes at an even faster rate; one may initially be inclined to increase R to improve the situation, but this would only worsen things! The minimum value of R_e corresponds to the minimum value of R. However, the minimum value of R must be constrained by the far-field conditions during transmission, or else some of the assumptions fail to be true. Using $R_{min} = 2d^2/\lambda$, then

$$R_{e,\,min} = 16\,R_{min}.$$

Apparently, the radar, when receiving the reflected energy, is never in the far field of the reradiated energy.

One may inquire as to whether the artificial constraint that R be greater than R_{min} (which was made to keep the analysis simple) may cause certain salient aspects of the problem to be overlooked. In particular, can it happen that R_e is ever less than R if this restriction is relaxed? This question is not answered here because in almost all practical radar geometries, the far-field condi-

tions during transmission are met by a wide margin.

The above result has several ramifications. The near field of a radiating surface is characterized by having wavefront normal vectors which do not necessarily point back at the center of that surface. (In angle-measuring radars, this is commonly referred to as *glint noise*.) Therefore, a radar which is being used to detect the direction of targets in its field of view, if such targets fill a large part of the radar's beam, may encounter difficulty. Such behavior is sometimes observed, for example, during end-game tactics of a missile seeker working against an airborne target.

The effects of this phenomenon on SAR, its relationship to speckle, and the possible apparent motion of stationary targets due to wobbling of the wavefront normal vector under look angle variation are areas for further investigation. An interesting investigation would be to find the image of two point scatterers under narrow-band excitation; if the aforementioned apparent motion due to wavefront anomalies is present in significant amounts, one would expect image degradation.

APPENDIX C

A RELATIONSHIP BETWEEN FAN-BEAM TOMOGRAPHY AND SAR

EMPLOYING simple geometric considerations, it will be shown in this appendix that a kind of fan-beam geometry exists when a uniformly rotating object is imaged by a SAR or similar imaging system. It is understood that the motion between the radar and the ground patch is continuous during transmission and reception and the intervening period. The degree of the effect will be seen to be very slight under most operating scenarios, but could become important in some cases.

The pertinent geometry is shown in Fig. C. 1. For convenience, assume that an impulsive plane wave is transmitted and that at time $t = 0$ it coincides exactly with the y axis. Further, let there be a coordinate system (x', y') rotating at a rate Ω rad/s and such that $x' = x$ when $t = 0$. The x'-axis always points towards the distant radar.

The crux of the situation is that since the motion between the radar and the ground patch spans the time that the transmitted wave and its reflections are in flight, the contours of equal times-of-flight are no longer parallel lines relative to the (x, y) system. In Fig. C. 2 are shown the approximate lines of equal times-of-flight of the impulsive wave as it passes over the ground patch, as would be seen by a radar with relative motion described by the rotating (x', y') coordinates. Each contour is a straight line, but they are splayed as shown in the figure. With $\theta = \Omega t$ and coordinate transformations

$$x' = x\cos\theta + y\sin\theta \qquad y' = -x\sin\theta + y\cos\theta$$

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

the image as a function of time is

$$g(x, y, t) = g\left[x'\cos(\Omega t) - y'\sin(\Omega t), x'\sin(\Omega t) + y'\cos(\Omega t)\right].$$

Now, the equation of a line in the (x', y')-plane is

$$y' = mx' + b$$

where b is the y'-intercept and where

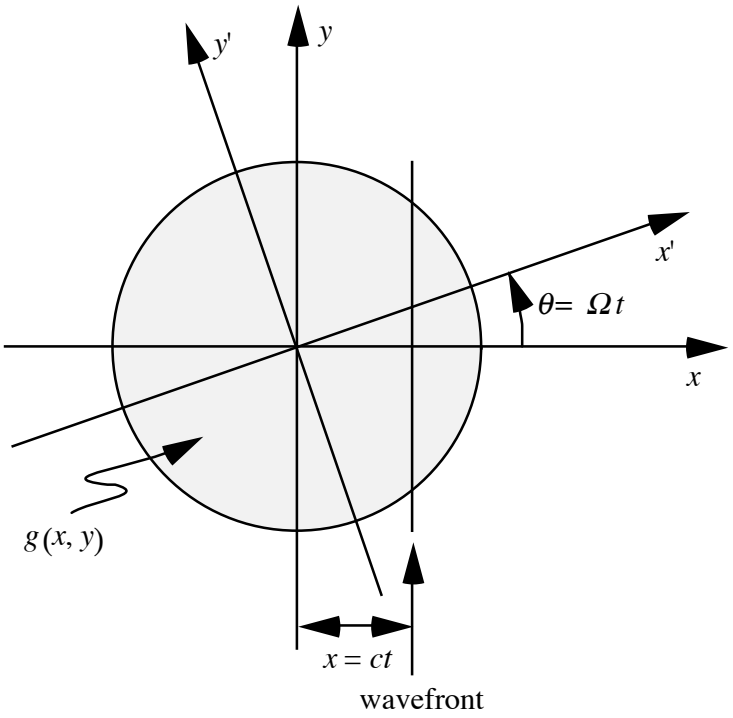


Fig. C. 1. Geometry for deriving a fan-beam-like relationship for SAR.

$$m = \frac{1}{\tan \theta}$$

The line described here is taken to be that of a wavefront at time t. The value of b is the point on the y' axis at which the wavefront crosses. Solving for b,

$$b = y' - mx'$$

$$= -x \sin \theta + y \cos \theta - \frac{\cos \theta \left(x \cos \theta + y \sin \theta \right)}{\sin \theta}$$

$$= -\frac{ct}{\sin(\Omega t)}. \qquad \text{(C. 1)}$$

Since b is not a constant with respect to time, there is not a strict fan beam relationship, the intercepts varying and not all intersecting at the same point. However, under a wide range of situations, the condition

$$\Omega t_{max} = \frac{\Omega L}{c} << 1 \qquad \text{(C. 2)}$$

where L is the patch radius, holds. With this,

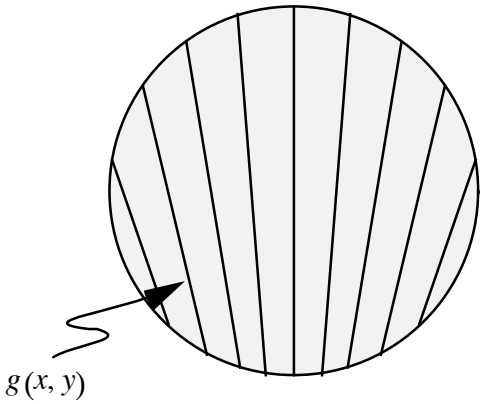


Fig. C. 2. An approximate display of fan-beam rays due to uniform relative motion between the radar and the ground patch.

$$b \approx -\frac{c}{\Omega} . \tag{C. 3}$$

This approximation is independent of time and holds over a reasonable range of rotation rates and patch sizes. This value of b is seen to be very large for almost all practical cases, but as calculated in the text, cannot be easily dismissed in all cases.

To the extent that approximation (C. 3) holds, the contours of equal times-of-flight constitute a fan beam, albeit a very narrow one in most applications. If (C. 3) does not apply, then (C. 1) indicates that a kind of modified fan beam exists, one which does not have a common center for all of its rays.

The above essentially completes the main point of this appendix; however, a slight expansion and generalization may prove useful. Referring to Fig. C. 3, let

$$\theta_1 = \Omega t_1$$

$$\theta_2 = \Omega t_2 .$$

Let two lines, representing two equal times-of-flight contours, be

$$y' = m_1 u + b_1$$

$$y' = m_2 u + b_2$$

with solution

$$x_0' = \frac{b_2 - b_1}{m_1 - m_2} \quad , \qquad y_0' = m_1 u_0 + b_1 .$$

Then their intersection at (x_0', y_0') is

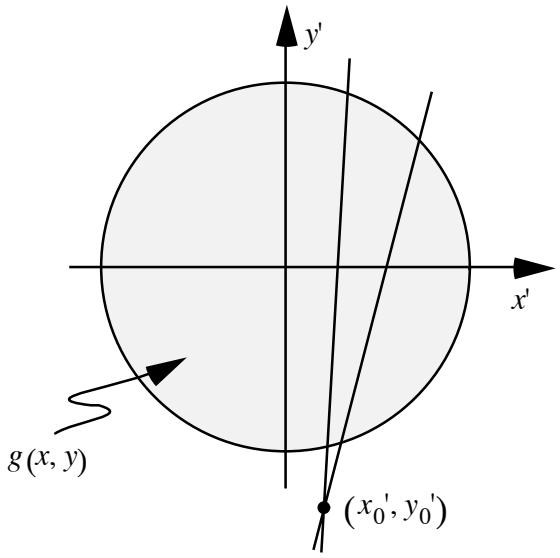


Fig. C. 3. Geometry for generalizing the original result.

$$x_0' = \frac{\dfrac{ct_1}{\sin\theta_1} - \dfrac{ct_2}{\sin\theta_2}}{\dfrac{1}{\tan\theta_1} - \dfrac{1}{\tan\theta_2}}$$

$$= \frac{ct_1 \sin\theta_2 - ct_2 \sin\theta_1}{\sin\theta_2 \cos\theta_1 - \sin\theta_1 \cos\theta_2}$$

$$= \frac{ct_2 \sin\theta_1 - ct_1 \sin\theta_2}{\sin(\theta_1 - \theta_2)} .$$

Note that if $t_2 = -t_1$, one would expect x_0' to be 0, by a symmetry argument. Also, if $t_1 = 0$ or $t_2 = 0$, one would again expect x_0' to be 0. Both of these expectations are supported by the above equation. Continuing,

$$y_0' = m_1 u_0 + b_1$$

$$= \frac{\cos\theta_1 (ct_2 \sin\theta_1 - ct_1 \sin\theta_2) - ct_1 \sin(\theta_1 - \theta_2)}{\sin\theta_1 \sin(\theta_1 - \theta_2)}$$

$$= \frac{ct_2 \cos\theta_1 \sin\theta_1 - ct_1 \cos\theta_1 \sin\theta_2 - ct_1 \sin(\theta_1 - \theta_2)}{\sin\theta_1 \sin(\theta_1 - \theta_2)}$$

$$= \frac{\dfrac{ct_2}{2}\sin(2\theta_1) - \dfrac{ct_1}{2}\left[\sin(\theta_1 + \theta_2) - \sin(\theta_1 - \theta_2)\right] - ct_1 \sin(\theta_1 - \theta_2)}{\sin\theta_1 \sin(\theta_1 - \theta_2)}$$

$$= \frac{\frac{ct_2}{2}\sin(2\theta_1) - \frac{ct_1}{2}\left[\sin(\theta_1 + \theta_2) + \sin(\theta_1 - \theta_2)\right]}{\sin\,\theta_1\ \sin(\theta_1 - \theta_2)}$$

$$= \frac{\frac{ct_2}{2}\sin(2\theta_1) - ct_1 \sin\,\theta_1\ \cos\,\theta_2}{\sin\,\theta_1\ \sin(\theta_1 - \theta_2)}$$

$$= \frac{ct_2 \sin\,\theta_1 \cos\,\theta_1 - ct_1 \sin\,\theta_1 \cos\,\theta_2}{\sin\,\theta_1\ \sin(\theta_1 - \theta_2)}$$

$$= \frac{ct_2 \cos\,\theta_1 - ct_1 \cos\,\theta_2}{\sin(\theta_1 - \theta_2)}\ .$$

Note that if $t_1 = 0$, this equation gives $y_0' = 0$ as expected. The polar coordinates of the point (x_0', y_0') can be found to be

$$r_0 = \frac{c\left[t_1^2 + t_2^2 - 2t_1 t_2 \cos(\theta_1 - \theta_2)\right]^{1/2}}{\sin(\theta_1 - \theta_2)}$$

and

$$\psi_0 = \tan^{-1}\left(\frac{ct_2 \cos\,\theta_1 - ct_1 \cos\,\theta_2}{ct_2 \sin\,\theta_1 - ct_1 \sin\,\theta_2}\right).$$

These results for x_0', y_0', r_0, and ψ_0 are all exact and as before they depend on time. Again using the approximation (C. 2), they simplify to

$$x_0' \approx 0$$

$$y_0' \approx -\frac{c}{\Omega}$$

$$r_0 \approx -\frac{c}{\Omega}$$

$$\psi_0 \approx -\frac{\pi}{2}\ .$$

Reconstruction algorithms for fan beams are in the literature, since the use of fan beams in X-ray tomography has been a way of reducing dosage to the patient and shortening the time that the patient must remain motionless. For this dissertation, algorithms which simply backproject the filtered projections along the original fan patterns were found to yield high-quality reconstructions. One algorithm assumed that the projection rays of the fan were evenly spaced in angle and the other algorithm assumed that they were spaced so as to be detected by a uniformly-spaced linear

array of detectors. Both algorithms were simpler than that of [36], which requires pre- and post-multiplication stages.

APPENDIX D

PROGRAM LISTINGS

LISTINGS of the source code used in the simulations of this dissertation are included here in order to form a complete record. Ordinarily, the reader would not be required to look at these, but if some question were to arise concerning some detail of how the simulations were conducted, this appendix is the definitive answer. Although rather lengthy, the programs included are the minimum set with respect to completeness.

The language used for the programs is THINK's Lightspeed Pascal, version 2.0, published by Symantec Corporation for the Apple Macintosh computer. Some of the programs were originally written in FORTRAN 77 using a compiler published by Absoft. Because of this, the Pascal listings herein invoke few features of the Pascal language that do not have some analog in FORTRAN, so conversion back to FORTRAN or comprehension by a FORTRAN programmer should not be difficult.

The programs that are included and a brief description of their function follows.

SLP (page 151): Calculates the Straight-Line Projections (Radon transform) of the test function.

CBP (page 154): Reconstructs the image using standard Convolution Back-Projection.

CAP (page 158): Calculates the Circular-Arc Projections of the test function.

CACBP (page 163): Reconstructs the image using Circular-Arc Convolution Back-Projection.

CACBP8 (page 170): Same as the above only exploiting eight-fold symmetry.

EAP (page 181): Calculates the Elliptical-Arc Projections, with or without propagation attenuation.

EACBPU (page 191): Reconstructs the image using Elliptical-Arc Convolution Back-Projection without correcting for propagation attenuation.

EACBPA (page 197): Reconstructs the image using Elliptical-Arc Convolution Back-Projection while correcting for propagation attenuation.

```
program SLP (input, output);
{This program computes the Radon transform of the "top hat" function,}
{with or without antenna shading by a cosine pattern.}
    uses
      RealFunctions;
    const
```

```
    Nrays = 128; {Number of rays per projection, counting the unused one.}
    Nproj = 198; {Number of projections}
    AngularExtent = TwoPi;
type
    YesNo = (yes, no);
var
    i, j: longint;
    proj: array [-64..63] of extended; {for 127-pt. projections--element -64 is 0.0}
    r, theta, p, c, rsquared: extended;
    a0, a1, c0, c1, a0squared, a1squared, r0, theta0, r1, theta1: extended;
    a2, a3, c2, c3, a2squared, a3squared, r2, theta2, r3, theta3: extended;
    h0, h1, h2, h3: extended;
    y0p, y1p, y2p, y3p: extended;
    L, gamma, cnst: extended;
    AngleInc: extended;
    RadonFileName: string;
    RadonFile: file of real;
    AntennaWeighting: YesNo;
    TextRect: rect;

begin
L := Nrays / 2.0 - 1.0; {radius of ground patch}
gamma := 0.9; {"frequency" of cosine antenna pattern; gamma := 1.0 puts null at L.}
cnst := gamma * Pi / (2.0 * L);

SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Prepare output file.}
RadonFileName := NewFileName('File for Radon transform:');
if RadonFileName = '' then
    ExitToShell;
rewrite(RadonFile, RadonFileName);
write(RadonFile, Nrays); {aka npx}
write(RadonFile, Nproj); {aka npy}

{Ask if want antenna weighting function.}
write('Do you want the antenna weighting function? [yes/no] ');
readln(AntennaWeighting);

{Report the angular extent of the projections.}
writeln('Projections are from an angle of ', AngularExtent : 10 : 5);
AngleInc := AngularExtent / Nproj;

{Initialize the unused slot in the projection array.}
proj[-64] := 0.0;

{Set coordinates, radii, and heights of the top hats.}
r0 := 20.0; {0}
theta0 := 135.9 * RadiansPerDegree;
a0 := 35.0;
a0squared := a0 * a0;
h0 := 0.95;

r1 := 60.0; {1}
theta1 := 47.0 * RadiansPerDegree;
a1 := 2.0;
a1squared := a1 * a1;
h1 := 0.75;
```

```
r2 := 10.0; {2}
theta2 := 135.9 * RadiansPerDegree;
a2 := 20.0;
a2squared := a2 * a2;
h2 := -0.2;

r3 := 40.0; {3}
theta3 := -89.1 * RadiansPerDegree;
a3 := 5.0;
a3squared := a3 * a3;
h3 := 0.5;

{begin loop over angle--Nproj projections between 0 and AngularExtent.}
for i := 1 to Nproj do
    begin {First one is the projection from angle 0.}
    writeln(i); {Monitor progress.}
    theta := (i - 1.0) * AngularExtent / Nproj; {Stop short of pi.}
    c0 := r0 * cos(theta0 - theta); {aka x0p}
    c1 := r1 * cos(theta1 - theta);
    c2 := r2 * cos(theta2 - theta);
    c3 := r3 * cos(theta3 - theta);

{Zero the projection array for later accumulation.}
    for j := -63 to 63 do
        proj[j] := 0.0;

{Compute each projection if don't want antenna weighting.}
    if AntennaWeighting = no then
        begin
        for j := -63 to 63 do
            begin
            p := j;
            if abs(p - c0) <= a0 then
                proj[j] := 2.0 * h0 * sqrt(a0squared - sqr(p - c0)); {height 0.95}
            if (abs(p - c1) <= a1) then
                proj[j] := proj[j] + 2.0 * h1 * sqrt(a1squared - sqr(p - c1)); {height 0.75}
            if (abs(p - c2) <= a2) then
                proj[j] := proj[j] + 2.0 * h2 * sqrt(a2squared - sqr(p - c2)); {height -0.2}
            if (abs(p - c3) <= a3) then
                proj[j] := proj[j] + 2.0 * h3 * sqrt(a3squared - sqr(p - c3)) {height 0.5}
            end;
        end
    else

{Compute each projection if do want antenna weighting.}
        begin
        y0p := r0 * sin(theta0 - theta);
        y1p := r1 * sin(theta1 - theta);
        y2p := r2 * sin(theta2 - theta);
        y3p := r3 * sin(theta3 - theta);
        for j := -63 to 63 do
            begin
            p := j;
            if abs(p - c0) <= a0 then
                proj[j] := 2.0 * h0 / cnst * cos(cnst * y0p) * sin(cnst * sqrt(a0squared - sqr(p - c0)));
            if (abs(p - c1) <= a1) then
                proj[j] := proj[j] + 2.0 * h1 / cnst * cos(cnst * y1p) * sin(cnst * sqrt(a1squared - sqr(p - c1)));
            if (abs(p - c2) <= a2) then
                proj[j] := proj[j] + 2.0 * h2 / cnst * cos(cnst * y2p) * sin(cnst * sqrt(a2squared - sqr(p - c2)));
            if (abs(p - c3) <= a3) then
```

```
                proj[j] := proj[j] + 2.0 * h3 / cnst * cos(cnst * y3p) * sin(cnst * sqrt(a3squared - sqr(p - c3)));
            end;
        end;

{Provide for the delta in angle and write each projection to a file.}
    for j := -64 to 63 do
        write(RadonFile, proj[j] * AngleInc);
    end;

write('Done.');
readln;
end.
```

```
program CBP (input, output);
{Convolution-backprojection along straight-line paths.}
    uses
        SANE, RealFunctions, ComplexFunctions, FFT256, ShutDownManager;
    label
        99;
    const
        RaysPerProj = 128;
        ImSize = 128; {Square image}
        RaysPerProjMinusOne = RaysPerProj - 1;
        RaysPerProjTimesTwo = RaysPerProj * 2;
        NumFFTPoints = RaysPerProjTimesTwo;
        ImSizeOv2 = ImSize div 2;
        ImSizeOv2Min1 = ImSizeOv2 - 1;
    type
        Row = array[-ImSizeOv2..ImSizeOv2] of real;              {A}
        RowPtr = ^Row;                                           {large}
        ImageArray = array[-ImSizeOv2..ImSizeOv2] of RowPtr; {array}
        YesNo = (yes, no);
    var
        Proj: array[1..RaysPerProj] of extended; {Projection}
        fProj: array[-RaysPerProj..RaysPerProjMinusOne] of extended; {Filtered projection}
        image: ImageArray; {the reconstructed image, indexed as image[row]^[col]}
        h: FFT256array;  {filter impulse response}
        clProj: FFT256array;  {Zero-padded complex version of Proj}
        k, i, j, npx, npy, mm, it, npxo2, BegTime, FinTime: longint;
        rnpx, rnpy, t, costheta, sintheta, xcostheta, ysintheta, jj, rit: extended;
        AngleMax: extended; {maximum angle from which projections are taken}
        PixelSkip, PixelCount: longint; {for low-resolution plots}
        WantWindow, quit, UpdateFile, WantShutDown: YesNo;
        InputAngularExtent: (PiOne, PiTwo);
        inFileName, outFileName: string;
        inFile, outFile: file of real;
        TextRect: rect;

    procedure Pause (PauseMessage: string);
    begin
    write(PauseMessage);
    readln;
    end;

    procedure InitArray (var TheArray: ImageArray;
                                    NumRows: longint;
                                    NumCols: longint);
    {Allocates memory from the heap for a large array.}
```

```
        var
            i: longint;
    begin
    for i := -NumRows div 2 to NumRows div 2 do
        TheArray[i] := RowPtr(NewPtr(SizeOf(real) * (NumCols + 1)));
    end;

begin{CBP}
InitArray(image, ImSize, ImSize);
SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Establish file connections.}
inFileName := OldFileName('File for projections:');
if inFileName = '' then
    ExitToShell;
outFileName := NewFileName('File for reconstruction:');
if outFileName = '' then
    ExitToShell;
reset(inFile, inFileName);

{Select image resolution parameter.}
write('Enter N, for skipping every Nth image pixel: ');
readln(PixelSkip);
if PixelSkip < 1 then
    begin
    pause('N must be >= 1.  Quitting.');
    ExitToShell
    end;

{Set up windowing option--Hamming or none.}
write('Do you want a smoothing window? [yes/no] ');
readln(WantWindow);

{Optional shutdown when the program is done.}
write('Do you want to shut down when the program is done? [yes/no] ');
readln(WantShutDown);

{Get data sizes.}
read(inFile, rnpx); {Number of rays per projection}
read(inFile, rnpy); {Number of projections}
npx := round(rnpx);
npy := round(rnpy);
npxo2 := npx div 2;

{Establish angular extent from which projections were taken.}
write('Angluar extent of the input projection data: [PiOne/PiTwo] ');
readln(InputAngularExtent);
if InputAngularExtent = PiOne then
    AngleMax := pi
else
    AngleMax := TwoPi;

GetDateTime(BegTime);
mm := 1 + round(log10(rnpx) / log10(2.0)); {FFT "log size"}

{Prepare frequency response array, for later filtering--see Haykin, p. 392.}
h[npxo2 + 1] := cmplx(0.125, 0.0); {n = npx/2 (64)--for AngularExtent = TwoPi}
if InputAngularExtent = PiOne then
    h[npxo2 + 1] := c(cmplx(2.0, 0.0), m, h[npxo2 + 1]); {for InputAngularExtent = Pi}
```

```
j := npxo2 + 3;
while j <= npx - 1 do { n:=66..126, even}
    begin
    h[j] := cmplx(0.0, 0.0);
    h[npx + 2 - j] := cmplx(0.0, 0.0); {n:=62..2, even}
    j := j + 2
    end;
h[1] := cmplx(0.0, 0.0); {n:=0}
j := npxo2 + 2;
while j <= npx do {n:=65..127, odd}
    begin
    jj := j - npxo2 - 1;
    h[j] := cmplx(-0.5 / (sqr(jj) * sqr(Pi)), 0.0); {for InputAngularExtent = TwoPi}
    if InputAngularExtent = PiOne then
        h[j] := c(cmplx(2.0, 0.0), m, h[j]); {for InputAngularExtent = Pi}
    h[npx + 2 - j] := h[j]; {n:=63..1, n odd}
    j := j + 2
    end;

{Zero pad the impulse response array.}
for j := npx + 1 to 2 * npx do
    h[j] := cmplx(0.0, 0.0);

FFT256(h, 1.0);

{Apply the optional smoothing window (Hamming).}
if WantWindow = yes then
    for j := 1 to 2 * npx do
        h[j] := c(h[j], m, cmplx((0.54 - 0.46 * cos(2.0 * Pi * (j - 1 - npx) / (2.0 * npx))), 0.0));

{Initialize the reconstructed-image array—code outliers with -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if sqrt(i * i + j * j) <= ImSizeOv2Min1 then
            image[i]^[j] := 0.0
        else
            image[i]^[j] := -INF;

{Begin loop over all filtered projections.}
for k := 0 to npy - 1 do {First one is the projection from angle 0.}
    begin
    writeln('Projection number ', k : 1); {Monitor progress}
    sintheta := sin(k * AngleMax / npy); {Stop short of AngleMax}
    costheta := cos(k * AngleMax / npy);

    {Get a projection from the input file}
    for j := 1 to npx do
        read(inFile, Proj[j]);

    {Make zero-padded, complex version of the projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);

    {Filter the projections.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);
```

```
    {Convert the double-length filtered projection to real form,}
    {and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj[j - npx - 1] := clProj[j].re;

    {Double loop over all pixels.}
    i := -ImSizeOv2;
    while i <= ImSizeOv2 do
        begin
        ysintheta := i * sintheta; {y:=float(i)}
        j := -ImSizeOv2;
        while j <= ImSizeOv2 do
            begin
            if image[i]^[j] <> -INF then {not outside the reconstruction circle.}
                begin
                xcostheta := j * costheta; {x:=float(j)}
                t := xcostheta + ysintheta;
                it := round(t - 0.5); {index into fProj, to begin linear interpolation}
                rit := it;

{Interpolate to t, between trunc(t) and trunc(t) + 1, and accumulate.}
                image[i]^[j] := image[i]^[j] + fProj[it] + (t - rit) * (fProj[it + 1] - fProj[it]);
                end;
            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [yes/no] ');
            readln(quit);
            if quit = yes then
                begin
                write('Do you want to update the output file? [yes/no] ');
                readln(UpdateFile);
                if UpdateFile = yes then
                    goto 99
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

{Fix the outlying pixels which were set to -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if image[i]^[j] = -INF then
            image[i]^[j] := 0.0;

{Prepare to write image to output file.}
99:
close(inFile);
rewrite(outFile, outFileName);

{Find how many non-zero pixels, in case of low-res.}
PixelCount := 0;
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    PixelCount := PixelCount + 1;
    i := i + PixelSkip
```

```
    end;
write(outFile, PixelCount); {Actual image size, in pixels.}
write(outFile, PixelCount);

{Write the image to the output file.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    j := -ImSizeOv2;
    while j <= ImSizeOv2 do
        begin
        write(outFile, image[i]^[j]);
        j := j + PixelSkip
        end;
    i := i + PixelSkip
    end;
close(outFile);

{Print out exit info to verify input selections.}
if WantWindow = yes then
    writeln('Hamming window');

writeln('Input file: ', inFileName);
writeln('Output file: ', outFileName);

GetDateTime(FinTime);
writeln('Execution time was: ', (FinTime - BegTime) / 60.0 : 5 : 1, ' minutes.');
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
if (WantShutDown = yes) and (not Button) then
    ShutDwnPower
else
    pause('Done');
end.
```

```
program CAP (input, output);
{This program computes the Circular-Arc Projections (CAPs) of}
{top hat test function.  Can simulate various flight trajectories.}
    uses
        SANE, RealFunctions;
{Dimension-dependent things.}
    const
        Nrays = 128; {Counting the unused one.}
        Nproj = 198;
        Nprojm1 = Nproj - 1;
        Nprojo2m1 = Nproj div 2 - 1;
        BigTheta1 = 0.1001674; {for Nproj of 200.  See notes, 5-20-87.}
        u = 0.1005038; {u is vxT/2xY0, notes, 5-22-87.}
    var
        proj: array[-64..63] of extended; {for 127-pt. projections--element -64 is 0.0}
        R0ar: array[0..Nprojm1] of extended; {array for variable radar radius.}
        Theta0ar: array[0..Nprojm1] of extended; {array for variable radar angle.}
        dTheta0ar: array[0..Nprojm1] of extended; {array for differential of radar angle.}
{end of dimension-dependent things.}
    var
```

```
        i, j: integer;
        r0, r1, theta0, theta1, r, theta, p, rsquared: extended;
        r2, r3, theta2, theta3: extended;
        a0, a0squared, a1, a1squared: extended;
        a2, a2squared, a3, a3squared: extended;
        x0, y0, x1, y1: extended;
        x2, y2, x3, y3: extended;
        AngleInc, AA, SS: extended;
        RR0, XX0, YY0, InitX0, vT: extended; {vT is velocity x PRF.}
        r0Hat, r0Hatsquared, acosarg0, r1Hat, r1Hatsquared, acosarg1: extended;
        r2Hat, r2Hatsquared, acosarg2, r3Hat, r3Hatsquared, acosarg3: extended;
        FlightType: char;
        FileName: string;
        outFile: file of real;
        TextRect: rect;

    procedure pause (PauseMessage: string);
    begin
    writeln(PauseMessage);
    readln
    end;

begin
SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Prepare output file.}
FileName := NewFileName('File for projections:');
rewrite(outFile, FileName);
write(outFile, Nrays); {aka npx}
write(outFile, Nproj); {aka npy}

{Initialize the unused slot in the projection array.}
proj[-64] := 0.0;

{Select the flight trajectory.}
writeln('Enter choice for flight trajectory.');
write('Dual fly-by, uniform Circle, Uniform-s fly-by, Square [d/c/u/s]: ');
readln(FlightType);

{Set up the arrays for radar location, in RR0, Theta0, dTheta0.}
if FlightType = 'c' then {Uniform circle}
    begin
    write('Enter RR0: ');
    readln(RR0);
    AngleInc := 2.0 * pi / Nproj;
    for i := 0 to Nprojm1 do
        begin
        R0ar[i] := RR0;
        Theta0ar[i] := i * AngleInc;
        dTheta0ar[i] := AngleInc;
        end
    end

else if (FlightType = 'd') then {Dual fly-by}
    begin

{Test that the number of projections is divisible by 4.}
    if Nproj mod 4 <> 0 then
        begin
```

```
        pause('Number of projections not divisible by 4.  Quitting.');
        ExitToShell
        end;

    write('Enter YY0: ');
    readln(YY0);
    InitX0 := YY0 / tan(BigTheta1); {for min max gap in sampling.}
    vT := u * 2.0 * YY0;
    for i := 0 to Nprojo2m1 do
        begin
        XX0 := InitX0 - i * vT;
        R0ar[i] := sqrt(sqr(XX0) + sqr(YY0)); {Out...}
        R0ar[Nprojm1 - i] := R0ar[i];   {and back.}
        Theta0ar[i] := arctan2(YY0, XX0);          {Out...}
        Theta0ar[Nprojm1 - i] := arctan2(-YY0, XX0); {and back.}
        dTheta0ar[i] := sqr(YY0 / sqr(R0ar[i])); {v:=1--Out...}
        dTheta0ar[Nprojm1 - i] := sqr(YY0 / sqr(R0ar[Nprojm1 - i])) {and back.}
        end;

{Fix the "U-turn" values for dTheta--same as at broadside.}
    dTheta0ar[0] := dTheta0ar[Nproj div 4];
    dTheta0ar[Nproj div 2 - 1] := dTheta0ar[Nproj div 4];
    dTheta0ar[Nproj div 2] := dTheta0ar[Nproj div 4];
    dTheta0ar[Nproj - 1] := dTheta0ar[Nproj div 4]
    end

else if FlightType = 'u' then {Dual fly-by}
    begin

{Test that the number of projections is divisible by 4.}
    if Nproj mod 4 <> 0 then
        begin
        pause('Number of projections not divisible by 4.  Quitting.');
        ExitToShell
        end;

    write('Enter YY0: ');
    readln(YY0);

{Set up theta and delta theta arrays.}
    for i := 0 to Nprojm1 do
        begin
        Theta0ar[i] := i * 2.0 * pi / Nproj;
        dTheta0ar[i] := 2.0 * pi / Nproj
        end;

{Set up range array.}
    for i := 1 to Nprojo2m1 do {1..99}
        begin
        R0ar[i] := YY0 / sin(Theta0ar[i]); {1..99}
        R0ar[Nproj - i] := R0ar[i]   {199..101}
        end;
    R0ar[0] := 1.0e4; {Some big number.  0}
    R0ar[Nproj div 2] := 1.0e4 { 100}
    end

else if FlightType = 's' then {Square}
    begin

{Test that the number of projections is divisible by 8.}
    if Nproj mod 8 <> 0 then
```

```
            begin
            pause('Number of projections not divisible by 8.  Quitting.');
            ExitToShell
            end;
        write('Enter AA--analogous to YY0: ');
        readln(AA);
        SS := 8.0 * AA / Nproj;

    {Set up theta and range arrays.}
        for i := 0 to Nproj div 8 do
            begin
            XX0 := AA;
            YY0 := i * SS;
            R0ar[i] := sqrt(sqr(XX0) + sqr(YY0));
            Theta0ar[i] := arctan2(YY0, XX0)
            end;
        for i := Nproj div 8 + 1 to (3 * Nproj) div 8 do
            begin
            XX0 := AA - (i - Nproj / 8.0) * SS;
            YY0 := AA;
            R0ar[i] := sqrt(sqr(XX0) + sqr(YY0));
            Theta0ar[i] := arctan2(YY0, XX0)
            end;
        for i := (3 * Nproj) div 8 + 1 to (5 * Nproj) div 8 do
            begin
            XX0 := -AA;
            YY0 := AA - (i - 3.0 * Nproj / 8.0) * SS;
            R0ar[i] := sqrt(sqr(XX0) + sqr(YY0));
            Theta0ar[i] := arctan2(YY0, XX0)
            end;
        for i := (5 * Nproj) div 8 + 1 to (7 * Nproj) div 8 do
            begin
            XX0 := -AA + (i - 5.0 * Nproj / 8.0) * SS;
            YY0 := -AA;
            R0ar[i] := sqrt(sqr(XX0) + sqr(YY0));
            Theta0ar[i] := arctan2(YY0, XX0)
            end;
        for i := (7 * Nproj) div 8 + 1 to Nprojm1 do
            begin
            XX0 := AA;
            YY0 := -AA + (i - 7.0 * Nproj / 8.0) * SS;
            R0ar[i] := sqrt(sqr(XX0) + sqr(YY0));
            Theta0ar[i] := arctan2(YY0, XX0)
            end;

    {Set up delta theta array.}
        dTheta0ar[0] := Theta0ar[1] - Theta0ar[0]; {0}
        for i := 1 to Nproj div 2 - 1 do {1..99}
            begin
            dTheta0ar[i] := (Theta0ar[i + 1] - Theta0ar[i - 1]) / 2.0;
            dTheta0ar[Nproj - i] := dTheta0ar[i]
            end;
        dTheta0ar[Nproj div 2] := dTheta0ar[0]
        end
    else
        begin
        pause('Bad input.  Hit RETURN to quit.');
        ExitToShell
        end;

    {Set coordinates of the top hats.}
```

```
r0 := 20.0; {0}
theta0 := 135.9 * RadiansPerDegree;
x0 := r0 * cos(theta0);
y0 := r0 * sin(theta0);

r1 := 60.0; {1}
theta1 := 47.0 * RadiansPerDegree;
x1 := r1 * cos(theta1);
y1 := r1 * sin(theta1);

r2 := 10.0; {2}
theta2 := 135.9 * RadiansPerDegree;
x2 := r2 * cos(theta2);
y2 := r2 * sin(theta2);

r3 := 40.0; {3}
theta3 := -89.1 * RadiansPerDegree;
x3 := r3 * cos(theta3);
y3 := r3 * sin(theta3);

{Set the radii of the top hats.}
a0 := 35.0;
a0squared := a0 * a0;

a1 := 2.0;
a1squared := a1 * a1;

a2 := 20.0;
a2squared := a2 * a2;

a3 := 5.0;
a3squared := a3 * a3;

{Compute CAP transform of the top hat function.}
{begin loop over angle--Nproj projections between 0 and 2*pi.}
for i := 0 to Nprojm1 do {First one is projection from angle 0.}
    begin
    writeln(i, Theta0ar[i] : 10 : 5); {Monitor progress.}

{Zero the projection array for later accumulation.}
    for j := -63 to 63 do
        proj[j] := 0.0;

{Find the radar's rectangular coordinates.}
    XX0 := R0ar[i] * cos(Theta0ar[i]);
    YY0 := R0ar[i] * sin(Theta0ar[i]);

    r0Hatsquared := sqr(x0 - XX0) + sqr(y0 - YY0);
    r0Hat := -sqrt(r0Hatsquared);

    r1Hatsquared := sqr(x1 - XX0) + sqr(y1 - YY0);
    r1Hat := -sqrt(r1Hatsquared);

    r2Hatsquared := sqr(x2 - XX0) + sqr(y2 - YY0);
    r2Hat := -sqrt(r2Hatsquared);

    r3Hatsquared := sqr(x3 - XX0) + sqr(y3 - YY0);
    r3Hat := -sqrt(r3Hatsquared);

{Compute each projection.}
    for j := -63 to 63 do
```

```
        begin

{Find the arc radius from the radar.}
        p := j;
        r := p - R0ar[i];
        rsquared := r * r;

        acosarg0 := (r0Hatsquared + rsquared - a0squared) / (2.0 * r0Hat * r);
        if abs(acosarg0) <= 1.0 then
            proj[j] := abs(1.9 * r * arccos(acosarg0)); {height 0.95}
        acosarg1 := (r1Hatsquared + rsquared - a1squared) / (2.0 * r1Hat * r);
        if abs(acosarg1) <= 1.0 then
            proj[j] := proj[j] + abs(1.5 * r * arccos(acosarg1)); {height 0.75}
        acosarg2 := (r2Hatsquared + rsquared - a2squared) / (2.0 * r2Hat * r);
        if abs(acosarg2) <= 1.0 then
            proj[j] := proj[j] - abs(0.4 * r * arccos(acosarg2)); {height -0.2}
        acosarg3 := (r3Hatsquared + rsquared - a3squared) / (2.0 * r3Hat * r);
        if abs(acosarg3) <= 1.0 then
            proj[j] := proj[j] + abs(1.0 * r * arccos(acosarg3)); {height 0.5}

{Weight each projection by the differential angle.  This}
{saves doing the weighting during the reconstruction.}
        proj[j] := proj[j] * dTheta0ar[i]
        end;

{Write each projection to a file.}
    for j := -64 to 63 do
        write(outFile, proj[j]);
    end;

{Print out exit info to verify flight trajectory selection, etc.}
if FlightType = 'c' then
    begin
    writeln('Flight trajectory is UNIFORM CIRCLE.');
    writeln('RR0 := ', R0ar[1] : 10 : 5)
    end
else if FlightType = 'd' then
    begin
    writeln('Flight trajectory is DUAL FLY-BY.');
    writeln('YY0 := ', YY0 : 10 : 5)
    end
else if FlightType = 'u' then
    begin
    writeln('Flight trajectory is UNIFORM-S DUAL FLY-BY.');
    writeln('YY0 := ', YY0 : 10 : 5)
    end
else if FlightType = 's' then
    begin
    writeln('Flight trajectory is SQUARE.');
    writeln('AA := ', AA : 10 : 5)
    end;
pause('Done.');
end.
```

```
program CACBP (input, output);
{Circular-Arc Convolution BackProjection}
{Simulates various flight trajectories.}
    uses
```

```
        SANE, RealFunctions, ComplexFunctions, FFT256;
    label
        99;
{Dimension-dependent things.}
    const
        RaysPerProj = 128;
        ImSize = 128;  {Square image size}
        RaysPerProjMinusOne = RaysPerProj - 1;
        RaysPerProjTimesTwo = RaysPerProj * 2;
        NumFFTPoints = RaysPerProjTimesTwo;
        ImSizeOv2 = ImSize div 2;
        ImSizeOv2Min1 = ImSizeOv2 - 1;
{end of dimension-dependent things.}
    const
        BigTheta1 = 0.1001674; {for Nproj of 200.}
        u = 0.1005038;   {u is vxT/2xY0.}
    type
    Row = array[-ImSizeOv2..ImSizeOv2] of real;                    {A}
        RowPtr = ^Row;                                              {large}
        ImageArray = array[-ImSizeOv2..ImSizeOv2] of RowPtr; {array}
        YesNo = (yes, no);
    var
        Proj: array [1..RaysPerProj] of extended; {Projection}
        fProj: array[-RaysPerProj..RaysPerProjMinusOne] of extended; {Filtered projection}
        image: ImageArray; {the reconstructed image, indexed as image[row]^[col]}
        R0ar: array[0..199] of extended; {array for variable radar radius.}
        Theta0ar: array[0..199] of extended; {array for variable radar angle.}
        h: FFT256array;  {filter impulse response}
        clProj: FFT256array;  {Zero-padded complex version of Proj}
        k, i, j, npx, npy, mm, it, npxo2: longint;
        BegTime, FinTime: longint;
        Nproj, Nprojm1, Nprojo2m1: longint;
        PixelSkip, PixelCount: longint; {for low-resolution plots.}
        rnpx, rnpy, costheta, sintheta, xcostheta, ysintheta, jj, rit: extended;
        xsintheta, ycostheta, R0: extended;
        xp, yp, xpp, rho, littletheta: extended; {Coordinate transformation stuff}
        AngleInc, X0, Y0, InitX0, vT, AA, SS: extended;
        WantWindow, quit, UpdateFile: YesNo;
        FlightType: char;
        inFileName, outFileName: string;
        inFile, outFile: file of real;
        TextRect: rect;

    procedure Pause (PauseMessage: string);
    begin
    write(PauseMessage);
    readln;
    end;

    procedure InitArray (var TheArray: ImageArray;
                                    NumRows: longint;
                                    NumCols: longint);
    {Allocates memory from the heap for a large array.}
        var
            i: longint;
    begin
    for i := -NumRows div 2 to NumRows div 2 do
        TheArray[i] := RowPtr(NewPtr(SizeOf(real) * (NumCols + 1)));
    end;

begin{CACBP}
```

```
InitArray(image, ImSize, ImSize);
SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Establish file connections.}
inFileName := OldFileName('Projection file:');
if inFileName <> '' then
    reset(inFile, inFileName)
else
    ExitToShell;

outFileName := NewFileName('File for reconstruction:');
if outFileName <> '' then
    rewrite(outFile, outFileName)
else
    ExitToShell;

{Select image resolution parameter.}
write('Enter N, for skipping every Nth image pixel: ');
readln(PixelSkip);
if PixelSkip < 1 then
    begin
    pause('N must be >= 1.  Quitting.');
    ExitToShell
    end;

{Get data sizes.}
read(inFile, rnpx); {Number of rays per projection}
read(inFile, rnpy); {Number of projections}
npx := round(rnpx);
npy := round(rnpy);
npxo2 := npx div 2;
Nproj := npy;
Nprojm1 := Nproj - 1;
Nprojo2m1 := Nproj div 2 - 1;

{Select the flight trajectory.}
writeln('Enter choice for flight trajectory.');
write('Dual fly-by, uniform Circle, Uniform-s fly-by, Square [d/c/u/s]: ');
readln(FlightType);

{Set up the arrays for projector location, in R0 and Theta0}
if FlightType = 'c' then {Uniform circle}
    begin
    write('Enter R0: ');
    readln(R0);
    AngleInc := 2.0 * Pi / Nproj;
    for i := 0 to Nprojm1 do
        begin
        R0ar[i] := R0;
        Theta0ar[i] := i * AngleInc
        end
    end
else if FlightType = 'd' then {Dual fly-by}
    begin
    write('Enter Y0: ');
    readln(Y0);
    InitX0 := Y0 / tan(BigTheta1); {for min max gap in sampling.}
    vT := u * 2.0 * Y0;
    for i := 0 to Nprojo2m1 do
```

```
        begin
        X0 := InitX0 - i * vT;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0)); {Out...}
        R0ar[Nprojm1 - i] := R0ar[i];  {and back.}
        Theta0ar[i] := arctan2(Y0, X0);         {Out...}
        Theta0ar[Nprojm1 - i] := arctan2(-Y0, X0) {and back.}
        end
    end
else if FlightType = 'u' then {Uniform dual fly-by}
    begin
    write('Enter Y0: ');
    readln(Y0);
    for i := 0 to Nprojm1 do
        Theta0ar[i] := i * 2.0 * Pi / Nproj;
    for i := 1 to Nprojo2m1 do
        begin
        R0ar[i] := Y0 / sin(Theta0ar[i]);
        R0ar[Nproj - i] := R0ar[i]
        end;
    R0ar[0] := 1.0e4; {Some really big number.}
    R0ar[Nproj div 2] := 1.0e4
    end
else if FlightType = 's' then {Square}
    begin

{Test that number of projections is divisible by 8.}
    if Nproj mod 8 <> 0 then
        begin
        pause('Number of projections is not divisible by 8.  Quitting.');
        ExitToShell
        end;

    write('Enter AA--analogous to Y0: ');
    readln(AA);
    SS := 8.0 * AA / Nproj;

{Set up theta and range arrays.}
    for i := 0 to Nproj div 8 do
        begin
        X0 := AA;
        Y0 := i * SS;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := Nproj div 8 + 1 to 3 * Nproj div 8 do
        begin
        X0 := AA - (i - Nproj / 8.0) * SS;
        Y0 := AA;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := (3 * Nproj) div 8 + 1 to (5 * Nproj) div 8 do
        begin
        X0 := -AA;
        Y0 := AA - (i - 3.0 * Nproj / 8.0) * SS;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := (5 * Nproj) div 8 + 1 to (7 * Nproj) div 8 do
        begin
        X0 := -AA + (i - 5.0 * Nproj / 8.0) * SS;
```

```
            Y0 := -AA;
            R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
            Theta0ar[i] := arctan2(Y0, X0)
            end;
        for i := (7 * Nproj) div 8 + 1 to Nprojm1 do
            begin
            X0 := AA;
            Y0 := -AA + (i - 7.0 * Nproj / 8.0) * SS;
            R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
            Theta0ar[i] := arctan2(Y0, X0)
            end
        end
    else
        begin
        pause('Bad input.  Hit RETURN to quit.');
        ExitToShell
        end;

    {Set up windowing option--Hamming or none.}
    write('Do you want a smoothing window? [yes/no] ');
    readln(WantWindow);

    SysBeep(1);
    GetDateTime(BegTime);
    mm := 1 + round(log10(rnpx) / log10(2.0)); {FFT "size"}

    {Prepare frequency response array, for later filtering--see Haykin, p. 392.}
    h[npxo2 + 1] := cmplx(0.125, 0.0); {n = npx/2 (64)}
    j := npxo2 + 3;
    while j <= npx - 1 do { n:=66..126, even}
        begin
        h[j] := cmplx(0.0, 0.0);
        h[npx + 2 - j] := cmplx(0.0, 0.0); {n:=62..2, even}
        j := j + 2
        end;
    h[1] := cmplx(0.0, 0.0); {n:=0}
    j := npxo2 + 2;
    while j <= npx do {n:=65..127, odd}
        begin
        jj := j - npxo2 - 1;
        h[j] := cmplx(-0.5 / (sqr(jj) * sqr(Pi)), 0.0);
        h[npx + 2 - j] := h[j]; {n:=63..1, n odd}
        j := j + 2
        end;

    {Zero pad the impulse response array.}
    for j := npx + 1 to 2 * npx do
        h[j] := cmplx(0.0, 0.0);

    FFT256(h, 1.0);

    {Apply the optional smoothing window (Hamming).}
    if WantWindow = yes then
        for j := 1 to 2 * npx do
            h[j] := c(h[j], m, cmplx((0.54 - 0.46 * cos(2.0 * Pi * (j - 1 - npx) / (2.0 * npx))), 0.0));

    {Initialize the reconstructed-image array—code outliers with -INF.}
    for i := -ImSizeOv2 to ImSizeOv2 do
        for j := -ImSizeOv2 to ImSizeOv2 do
            if sqrt(i * i + j * j) <= ImSizeOv2Min1 then
                image[i]^[j] := 0.0
```

```
        else
            image[i]^[j] := -INF;

{Begin loop over all filtered projections.}
for k := 0 to npy - 1 do {First one is the projection from angle 0.}
    begin
    writeln('Projection number ', k : 1); {Monitor progress}
    sintheta := sin(Theta0ar[k]);
    costheta := cos(Theta0ar[k]);
    R0 := R0ar[k];

{Get a projection from the input file}
    for j := 1 to npx do
        read(inFile, Proj[j]);

{Make zero-padded, complex version of the projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);

{Filter the projections.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);

{Convert the double-length filtered projection to real form,}
{and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj[j - npx - 1] := clProj[j].re;

{Double loop over all pixels.}
    i := -ImSizeOv2;
    while i <= ImSizeOv2 do
        begin
        ysintheta := i * sintheta; {y:=float(i)}
        ycostheta := i * costheta;

        j := -ImSizeOv2;
        while j <= ImSizeOv2 do
            begin
            if image[i]^[j] <> -INF then {not outside the reconstruction circle.}
                begin
                xcostheta := j * costheta; {x:=float(j)}
                xsintheta := j * sintheta;
                xp := xcostheta + ysintheta;
                yp := ycostheta - xsintheta;
                rho := sqrt(sqr(xp - R0) + sqr(yp));
                xpp := R0 - rho; {Don't need ypp.  Analogous to t in CBP.}
                it := round(xpp - 0.5); {index into fProj for linear interpolation}
                rit := it;

{Interpolate to t, between it and it + 1, and accumulate.}
                image[i]^[j] := image[i]^[j] + fProj[it] + (xpp - rit) * (fProj[it + 1] - fProj[it]);
                end;
            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [yes/no] ');
```

```
            readln(quit);
            if quit = yes then
                begin
                write('Do you want to update the output file? [yes/no] ');
                readln(UpdateFile);
                if UpdateFile = yes then
                    goto 99
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

{Fix the outlying pixels which were set to -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if image[i]^[j] = -INF then
            image[i]^[j] := 0.0;

{Prepare to write image to output file.}
99:
close(inFile);

{Find how many non-zero pixels, in case of low-res.}
PixelCount := 0;
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    PixelCount := PixelCount + 1;
    i := i + PixelSkip
    end;
write(outFile, PixelCount); {Actual image size, in pixels.}
write(outFile, PixelCount);

{Write the image to the output file.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    j := -ImSizeOv2;
    while j <= ImSizeOv2 do
        begin
        write(outFile, image[i]^[j]);
        j := j + PixelSkip
        end;
    i := i + PixelSkip
    end;
close(outFile);

{Print out exit info to verify flight trajectory selection, etc.}
if FlightType = 'c' then
    begin
    writeln('Flight trajectory is UNIFORM CIRCLE.');
    writeln('R0 = ', R0ar[1] : 10 : 5)
    end
else if FlightType = 'd' then
    begin
    writeln('Flight trajectory is DUAL FLY-BY.');
    writeln('Y0 = ', Y0 : 10 : 5)
    end
```

```
else if FlightType = 'u' then
    begin
    writeln('Flight trajectory is UNIFORM-S DUAL FLY-BY.');
    writeln('Y0 = ', Y0 : 10 : 5)
    end
else if FlightType = 's' then
    begin
    writeln('Flight trajectory is SQUARE.');
    writeln('AA = ', AA : 10 : 5)
    end;

if WantWindow = yes then
    writeln('Hamming window');

GetDateTime(FinTime);
writeln('Execution time was: ', (FinTime - BegTime) / 60.0 : 5 : 1, ' minutes.');
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
pause('Done');
end.
```

```
program CACBP8 (input, output);
{Circular-Arc Convolution BackProjection}
{Simulates various flight trajectories.}
{Takes advantage of eight-fold symmetry to increase efficiency.}
    uses
        SANE, RealFunctions, ComplexFunctions, FFT256;
    label
        99;
{Dimension-dependent things.}
    const
        RaysPerProj = 128;
        ImSize = 128;  {Square image size}
        RaysPerProjMinusOne = RaysPerProj - 1;
        RaysPerProjTimesTwo = RaysPerProj * 2;
        NumFFTPoints = RaysPerProjTimesTwo;
        ImSizeOv2 = ImSize div 2;
        ImSizeOv2Min1 = ImSizeOv2 - 1;
{end of dimension-dependent things.}
    const
        BigTheta1 = 0.1001674; {for Nproj of 200..}
        u = 0.1005038;   {u is vxT/2xY0.}
    type
        Row = array[-ImSizeOv2..ImSizeOv2] of real;              {A}
        RowPtr = ^Row;                                           {large}
        ImageArray = array[-ImSizeOv2..ImSizeOv2] of RowPtr; {array}
        YesNo = (y, n);
        fProjType = array[-RaysPerProj..RaysPerProjMinusOne] of extended;
    var
        Proj: array [1..RaysPerProj] of extended; {Projection (128)}
        fProj0, fProj1, fProj2, fProj3: array[-RaysPerProj..RaysPerProjMinusOne] of extended; {Filtered projection (-
128:127)}
        fProj4, fProj5, fProj6, fProj7: ^fProjType; {Filtered projection (-128:127)--These are pointers because of 32K limit
on global space.}
        image: ImageArray; {Indexed as image[row]^[col] — this is the reconstructed image}
```

```
        R0ar: array[0..199] of extended; {array for variable radar radius.}
        Theta0ar: array[0..199] of extended; {array for variable radar angle.}
        h: FFT256array;  {filter impulse response (256)}
        clProj: FFT256array;  {Zero-padded complex version of Proj}
        k, i, j, kk, npx, npy, mm, it, npxo2: longint;
        BegTime, FinTime: longint;
        Nproj, Nprojm1, Nprojo2m1: longint;
        PixelSkip, PixelCount: longint; {for low-resolution plots.}
        npydiv4, npydiv8: longint; {npy div 4, npy div 8}
        itPlus1: longint;
        rnpx, rnpy, costheta, sintheta, xcostheta, ysintheta, jj, xppMinusit: extended;
        xsintheta, ycostheta, R0: extended;
        xp, yp, xpp, rho, littletheta: extended; {Coordinate transformation stuff}
        AngleInc, X0, Y0, InitX0, vT, AA, SS: extended;
        WantWindow, quit, UpdateFile, ZeroClip: YesNo;
        FlightType: char;
        inFileName, outFileName: string;
        inFile, outFile: file of real;
        TextRect: rect;

    procedure Pause (PauseMessage: string);
    begin
    write(PauseMessage);
    readln;
    end;

    procedure InitArray (var TheArray: ImageArray;
                                    NumRows: longint;
                                    NumCols: longint);
    {Allocates memory from the heap for a large array.}
        var
            i: longint;
    begin
    for i := -NumRows div 2 to NumRows div 2 do
        TheArray[i] := RowPtr(NewPtr(SizeOf(real) * (NumCols + 1)));
    end;

begin{CACBP8}
InitArray(image, ImSize, ImSize);

{Allocate memory for four of the filtered projection arrays.}
new(fProj4);
new(fProj5);
new(fProj6);
new(fproj7);

SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Establish file connections.}
inFileName := OldFileName('Projection file:');
if inFileName <> '' then
    reset(inFile, inFileName)
else
    ExitToShell;

outFileName := NewFileName('File for reconstruction:');
if outFileName <> '' then
    rewrite(outFile, outFileName)
else
```

```
    ExitToShell;

{Select image resolution parameter.}
write('Enter N, for skipping every Nth image pixel: ');
readln(PixelSkip);
if PixelSkip < 1 then
    begin
    pause('N must be >= 1.  Quitting.');
    ExitToShell
    end;

{Get data sizes.}
read(inFile, rnpx); {Number of rays per projection}
read(inFile, rnpy); {Number of projections}
npx := round(rnpx);
npy := round(rnpy);
npxo2 := npx div 2;
Nproj := npy;
Nprojm1 := Nproj - 1;
Nprojo2m1 := Nproj div 2 - 1;

{Select the flight trajectory.}
writeln('Enter choice for flight trajectory.');
write('Dual fly-by, uniform Circle, Uniform-s fly-by, Square [d/c/u/s]: ');
readln(FlightType);

{Set up the arrays for radar location, in R0 and Theta0}
if FlightType = 'c' then {Uniform circle}
    begin
    write('Enter R0: ');
    readln(R0);
    AngleInc := TwoPi / Nproj;
    for i := 0 to Nprojm1 do
        begin
        R0ar[i] := R0;
        Theta0ar[i] := i * AngleInc
        end
    end
else if FlightType = 'd' then {Dual fly-by}
    begin
    write('Enter Y0: ');
    readln(Y0);
    InitX0 := Y0 / tan(BigTheta1); {for min max gap in sampling.}
    vT := u * 2.0 * Y0;
    for i := 0 to Nprojo2m1 do
        begin
        X0 := InitX0 - i * vT;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0)); {Out...}
        R0ar[Nprojm1 - i] := R0ar[i];  {and back.}
        Theta0ar[i] := arctan2(Y0, X0);         {Out...}
        Theta0ar[Nprojm1 - i] := arctan2(-Y0, X0) {and back.}
        end
    end
else if FlightType = 'u' then {Uniform dual fly-by}
    begin
    write('Enter Y0: ');
    readln(Y0);
    for i := 0 to Nprojm1 do
        Theta0ar[i] := i * 2.0 * Pi / Nproj;
    for i := 1 to Nprojo2m1 do {1..99}
        begin
```

```
        R0ar[i] := Y0 / sin(Theta0ar[i]); {1..99}
        R0ar[Nproj - i] := R0ar[i]   {199..101}
        end;
    R0ar[0] := 1.0e4; {Some really big number.  0}
    R0ar[Nproj div 2] := 1.0e4 {100}
    end
else if FlightType = 's' then {Square}
    begin

{Test that number of projections is divisible by 8.}
    if Nproj mod 8 <> 0 then
        begin
        pause('Number of projections not divisible by 8.  Quitting.');
        ExitToShell
        end;
    write('Enter AA--analogous to Y0: ');
    readln(AA);
    SS := 8.0 * AA / Nproj;

{Set up theta and range arrays.}
    for i := 0 to Nproj div 8 do
        begin
        X0 := AA;
        Y0 := i * SS;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := Nproj div 8 + 1 to 3 * Nproj div 8 do
        begin
        X0 := AA - (i - Nproj / 8.0) * SS;
        Y0 := AA;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := (3 * Nproj) div 8 + 1 to (5 * Nproj) div 8 do
        begin
        X0 := -AA;
        Y0 := AA - (i - 3.0 * Nproj / 8.0) * SS;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := (5 * Nproj) div 8 + 1 to (7 * Nproj) div 8 do
        begin
        X0 := -AA + (i - 5.0 * Nproj / 8.0) * SS;
        Y0 := -AA;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end;
    for i := (7 * Nproj) div 8 + 1 to Nprojm1 do
        begin
        X0 := AA;
        Y0 := -AA + (i - 7.0 * Nproj / 8.0) * SS;
        R0ar[i] := sqrt(sqr(X0) + sqr(Y0));
        Theta0ar[i] := arctan2(Y0, X0)
        end
    end
else
    begin
    pause('Bad input.  Hit RETURN to quit.');
    ExitToShell
    end;
```

```
{Set up windowing option--Hamming or none.}
write('Do you want a smoothing window? [y/n] ');
readln(WantWindow);

{Optional clipping of negative image values.}
write('Do you want to clip negative image values? [y/n] ');
readln(ZeroClip);

SysBeep(1);
SysBeep(1);
GetDateTime(BegTime);
mm := 1 + round(log10(rnpx) / log10(2.0)); {FFT "size"}

{Prepare frequency response array, for later filtering--see Haykin, p. 392.}
h[npxo2 + 1] := cmplx(0.125, 0.0); {n = npx/2 (64)}
j := npxo2 + 3;
while j <= npx - 1 do { n:=66..126, even}
    begin
    h[j] := cmplx(0.0, 0.0);
    h[npx + 2 - j] := cmplx(0.0, 0.0); {n:=62..2, even}
    j := j + 2
    end;
h[1] := cmplx(0.0, 0.0); {n:=0}
j := npxo2 + 2;
while j <= npx do {n:=65..127, odd}
    begin
    jj := j - npxo2 - 1;
    h[j] := cmplx(-0.5 / (sqr(jj) * sqr(Pi)), 0.0);
    h[npx + 2 - j] := h[j]; {n:=63..1, n odd}
    j := j + 2
    end;

{Zero pad the impulse response array.}
for j := npx + 1 to 2 * npx do
    h[j] := cmplx(0.0, 0.0);

FFT256(h, 1.0);

{Apply the optional smoothing window (Hamming).}
if WantWindow = y then
    for j := 1 to 2 * npx do
        h[j] := c(h[j], m, cmplx((0.54 - 0.46 * cos(TwoPi * (j - 1 - npx) / (2.0 * npx))), 0.0));

{Zero the reconstructed-image array.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        image[i]^[j] := 0.0;

{Make sure the number of projections is divisible by 8.}
if (npy mod 8) <> 0 then
    begin
    pause('Number of projections is not divisible by 8.  Hit RETURN to quit.');
    ExitToShell;
    end;
npydiv8 := npy div 8;
npydiv4 := npy div 4;

{Begin loop over all filtered projections.}
for k := 1 to npydiv8 - 1 do {First one is the projection from angle 0.}
    begin
```

```
writeln('Projection number ', k : 1); {Monitor progress}
sintheta := sin(Theta0ar[k]);
costheta := cos(Theta0ar[k]);
R0 := R0ar[k];

{Get zeroth projection from the input file.}
seek(inFile, 2 + (k) * npx); {1 to 24}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the zeroth projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the zeroth projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the zeroth double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj0[j - npx - 1] := clProj[j].re;

{Get first projection from the input file.}
seek(inFile, 2 + (npydiv4 - k) * npx); {49 down to 26}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the first projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the first projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the first double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj1[j - npx - 1] := clProj[j].re;

{Get second projection from the input file.}
seek(inFile, 2 + (npydiv4 + k) * npx); {51 to 74}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the first projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the first projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the first double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj2[j - npx - 1] := clProj[j].re;
```

```
{Get third projection from the input file.}
seek(inFile, 2 + (2 * npydiv4 - k) * npx); {99 down to 76}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the second projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the second projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the second double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj3[j - npx - 1] := clProj[j].re;

{Get fourth projection from the input file.}
seek(inFile, 2 + (2 * npydiv4 + k) * npx); {101 to 124}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the second projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the second projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the second double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj4^[j - npx - 1] := clProj[j].re;

{Get fifth projection from the input file.}
seek(inFile, 2 + (3 * npydiv4 - k) * npx); {149 down to 126}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the third projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the third projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the third double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj5^[j - npx - 1] := clProj[j].re;

{Get sixth projection from the input file.}
seek(inFile, 2 + (3 * npydiv4 + k) * npx); {151 to 174}
for j := 1 to npx do
```

```
        read(inFile, Proj[j]);
    {Make a zero-padded, complex version of the third projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);
    {Filter the third projection.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);
    {Convert the third double-length filtered projection to}
    { real form and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj6^[j - npx - 1] := clProj[j].re;

    {Get seventh projection from the input file.}
    seek(inFile, 2 + (4 * npydiv4 - k) * npx); {199 down to 176}
    for j := 1 to npx do
        read(inFile, Proj[j]);
    {Make a zero-padded, complex version of the zeroth projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);
    {Filter the zeroth projection.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);
    {Convert the zeroth double-length filtered projection to}
    { real form and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj7^[j - npx - 1] := clProj[j].re;

    {Double loop over all pixels.}
    i := -ImSizeOv2;
    while i <= ImSizeOv2 do
        begin
        ysintheta := i * sintheta; {y:=float(i)}
        ycostheta := i * costheta;

        j := -ImSizeOv2; {for j := -ImSizeOv2 to ImSizeOv2 by PixelSkip}
        while j <= ImSizeOv2 do
            begin
            xcostheta := j * costheta; {x:=float(j)}
            xsintheta := j * sintheta;
            xp := xcostheta + ysintheta;
            yp := ycostheta - xsintheta;
            rho := sqrt(sqr(xp - R0) + sqr(yp));
            xpp := R0 - rho; {Don't need ypp.  Analogous to t in CBP.}
            it := round(xpp - 0.5); {index into fProj for linear interpolation}
            xppMinusit := xpp - it;
            itPlus1 := it + 1;

{Interpolate to t, between it and it + 1, and accumulate.}
            image[i]^[j] := image[i]^[j] + fProj0[it] + xppMinusit * (fProj0[itPlus1] - fProj0[it]);
            image[j]^[i] := image[j]^[i] + fProj1[it] + xppMinusit * (fProj1[itPlus1] - fProj1[it]);

            image[j]^[-i] := image[j]^[-i] + fProj2[it] + xppMinusit * (fProj2[itPlus1] - fProj2[it]);
            image[i]^[-j] := image[i]^[-j] + fProj3[it] + xppMinusit * (fProj3[itPlus1] - fProj3[it]);
```

```
            image[-i]^[-j] := image[-i]^[-j] + fProj4^[it] + xppMinusit * (fProj4^[itPlus1] - fProj4^[it]);
            image[-j]^[-i] := image[-j]^[-i] + fProj5^[it] + xppMinusit * (fProj5^[itPlus1] - fProj5^[it]);

            image[-j]^[i] := image[-j]^[i] + fProj6^[it] + xppMinusit * (fProj6^[itPlus1] - fProj6^[it]);
            image[-i]^[j] := image[-i]^[j] + fProj7^[it] + xppMinusit * (fProj7^[itPlus1] - fProj7^[it]);

            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [y/n] ');
            readln(quit);
            if quit = y then
                begin
                write('Do you want to update the output file? [y/n] ');
                readln(UpdateFile);
                if UpdateFile = y then
                    goto 99 {Sorry about that.}
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

{Now do k = 0, npydiv4, 2*npydiv4, 3*npydiv4, and npydiv8, 3*npydiv8, 5*npydiv8, 7*npydiv8}
{by using two applications of four-fold symmetry.}
{Begin loop over all filtered projections.}
for kk := 1 to 2 do {First one is the projection from angle 0.}
    begin
    k := (kk - 1) * npydiv8; {0, 25}
    writeln('Projection number ', k : 1); {Monitor progress}
    sintheta := sin(Theta0ar[k]);
    costheta := cos(Theta0ar[k]);
    R0 := R0ar[k];

    {Get zeroth projection from the input file.}
    seek(inFile, 2 + k * npx); {0, 25}
    for j := 1 to npx do
        read(inFile, Proj[j]);
    {Make a zero-padded, complex version of the zeroth projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);
    {Filter the zeroth projection.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);
    {Convert the zeroth double-length filtered projection to}
    { real form and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj0[j - npx - 1] := clProj[j].re;

    {Get first projection from the input file.}
    seek(inFile, 2 + (npydiv4 + k) * npx); {50, 75}
    for j := 1 to npx do
        read(inFile, Proj[j]);
```

```
{Make a zero-padded, complex version of the first projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the first projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the first double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj1[j - npx - 1] := clProj[j].re;

{Get second projection from the input file.}
seek(inFile, 2 + (2 * npydiv4 + k) * npx); {100, 125}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the second projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the second projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the second double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj2[j - npx - 1] := clProj[j].re;

{Get third projection from the input file.}
seek(inFile, 2 + (3 * npydiv4 + k) * npx); {150, 175}
for j := 1 to npx do
    read(inFile, Proj[j]);
{Make a zero-padded, complex version of the third projection.}
for j := 1 to npx do
    clProj[j] := cmplx(Proj[j], 0.0);
for j := npx + 1 to 2 * npx do {zero padding}
    clProj[j] := cmplx(0.0, 0.0);
{Filter the third projection.}
FFT256(clProj, 1.0);
for j := 1 to 2 * npx do
    clProj[j] := c(clProj[j], m, h[j]);
FFT256(clProj, -1.0);
{Convert the third double-length filtered projection to}
{ real form and change indexing to image coordinates.}
for j := 1 to 2 * npx do
    fProj3[j - npx - 1] := clProj[j].re;

{Double loop over all pixels.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    ysintheta := i * sintheta; {y:=float(i)}
    ycostheta := i * costheta;

    j := -ImSizeOv2;
```

```
        while j <= ImSizeOv2 do
            begin
            xcostheta := j * costheta; {x:=float(j)}
            xsintheta := j * sintheta;
            xp := xcostheta + ysintheta;
            yp := ycostheta - xsintheta;
            rho := sqrt(sqr(xp - R0) + sqr(yp));
            xpp := R0 - rho; {Don't need ypp.  Analogous to t in CBP.}
            it := trunc(xpp); {index into fProj, to begin interpolation}
            xppMinusit := xpp - it;
            itPlus1 := it + 1;

{Interpolate to t, between trunc(t) and trunc(t) + 1, and accumulate.}
            image[i]^[j] := image[i]^[j] + fProj0[it] + xppMinusit * (fProj0[itPlus1] - fProj0[it]);
            image[j]^[-i] := image[j]^[-i] + fProj1[it] + xppMinusit * (fProj1[itPlus1] - fProj1[it]);
            image[-i]^[-j] := image[-i]^[-j] + fProj2[it] + xppMinusit * (fProj2[itPlus1] - fProj2[it]);
            image[-j]^[i] := image[-j]^[i] + fProj3[it] + xppMinusit * (fProj3[itPlus1] - fProj3[it]);

            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [y/n] ');
            readln(quit);
            if quit = y then
                begin
                write('Do you want to update the output file? [y/n] ');
                readln(UpdateFile);
                if UpdateFile = y then
                    goto 99
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

{Optionally zero-clip negative image values.}
if ZeroClip = y then
    for i := -ImSizeOv2 to ImSizeOv2 do
        for j := -ImSizeOv2 to ImSizeOv2 do
            if image[i]^[j] < 0.0 then
                image[i]^[j] := 0.0;

{Prepare to write image to output file.}
99:
close(inFile);
rewrite(outFile, outFileName);

{Find how many non-zero pixels, in case of low-res.}
PixelCount := 0;
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    PixelCount := PixelCount + 1;
    i := i + PixelSkip
    end;
write(outFile, PixelCount); {Actual image size, in pixels.}
write(outFile, PixelCount); {It's square.}
```

```
{Write the image to the output file.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    j := -ImSizeOv2;
    while j <= ImSizeOv2 do
        begin
        write(outFile, image[i]^[j]);
        j := j + PixelSkip;
        end;
    i := i + PixelSkip
    end;
close(outFile);

{Print out exit info to verify flight trajectory selection, etc.}
if FlightType = 'c' then
    begin
    writeln('Flight trajectory is UNIFORM CIRCLE.');
    writeln('R0 = ', R0ar[1] : 10 : 5)
    end
else if FlightType = 'd' then
    begin
    writeln('Flight trajectory is DUAL FLY-BY.');
    writeln('Y0 = ', Y0 : 10 : 5)
    end
else if FlightType = 'u' then
    begin
    writeln('Flight trajectory is UNIFORM-S DUAL FLY-BY.');
    writeln('Y0 = ', Y0 : 10 : 5)
    end
else if FlightType = 's' then
    begin
    writeln('Flight trajectory is SQUARE.');
    writeln('AA = ', AA : 10 : 5)
    end;

if WantWindow = y then
    writeln('Hamming window');

if ZeroClip = y then
    write('Zero-clipped');

GetDateTime(FinTime);
writeln('Execution time was: ', (FinTime - BegTime) / 60.0 : 1 : 1, ' minutes.');
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);

pause('Done');
end.
```

```
program EAP (input, output);
{This program computes various kinds of elliptical-arc projections of top hat functions.}
{It handles generalized radar trajectories.}
{The units referenced in the following "uses" clause called "rtbis" is a bisection method for finding}
{a root of an equation.  It is copyrighted material from "Numerical Recipes (see References) and so is not}
```

```
{included here.  The unit "el2" computes elliptic integrals and is also in "Numerical Recipes."  The units}
{"SolveCubic, SolveQuartic, and Solve Quadratic are closed-form solvers for polynomials.  The unit }
{"Plotter2 "is a subroutine to plot the transmitter and receiver trajectories.  These latter units are}
{not included in the interest of saving space.}
    uses
        SANE, RealFunctions, ComplexFunctions, SolveCubic, SolveQuartic, SolveQuadratic, el2, rtbis,
ShutDownManager, Plotter2;
    var
        AA0, B0, F0: extended; {ellipse params}
        ya, yb, yc, yd: extended;
        bigX0, bigY0, thetab, thetab0, costhetab, sinthetab: extended;
        initialR0: extended; {initial distance to the center of the bistatic system}
        Weighting: (NoWeight, Attenuation); {weighting for propagation attenuation}
        quit, WantShutDown: YesNo;
        TextRect, DrawingRect: rect;
    const
        small = 1.0e-14; {for function rtbis to quit}
        OrbModFreq = 5.0; {Generalized trajectory parameter:  Orbital Modulation Frequency}
        OrbModDepth = 0.25; {Orbital Modulation Depth}
        RotateRate = -1.0; {Rotation rate of the bistatic coordinates.}

{Dimension-dependent stuff.}
    const
        Nrays = 128; {Counting the unused one.}
        Nproj = 198; {198}
        Nprojm1 = Nproj - 1; {197}
        Nprojo2m1 = Nproj div 2 - 1; {98}

    var
        proj: array [-64..63] of extended; {for 127-pt. projections--element -64 is 0.0}
        Xt, Yt: array[0..Nprojm1] of extended; {coords of transmitter}
        Xr, Yr: array [0..Nprojm1] of extended; {coords of receiver}
        B0min, B0max: array[0..Nprojm1] of extended; {begin and end of sampling}
{end of dimension-dependent stuff.}

    var
        Monostatic: boolean;
        i, j: integer;
        r0, r1, theta0, theta1, r, theta, p, rsquared: extended;
        r2, r3, theta2, theta3: extended;
        a0, a1, a2, a3: extended; {These should be in arrays.}
        h0, h1, h2, h3: extended;
        x0, y0, x1, y1, x2, y2, x3, y3: extended;
        Xt0, Yt0, Xr0, Yr0, Angle: extended;
        Thetat0, Rt0, Thetar0, Rr0: extended;
        F, ksquared, kcsquared, kc: extended;
        B0scale: extended;
        B0minLo, B0minHi, B0minLast, B0maxLo, B0maxHi, B0maxLast: extended;
        AngleInc: extended;
        XX0, YY0: extended;
        r0Hat, r0Hatsquared, acosarg0, r1Hat, r1Hatsquared, acosarg1: extended;
        r2Hat, r2Hatsquared, acosarg2, r3Hat, r3Hatsquared, acosarg3: extended;
        InitialPosition: (Vertical, Horizontal, Oblique);
        xPlot, yPlot: Plot2array;
        FlightType: char;
        FileName, TrajectoryFileName: string;
        outFile, TrajectoryFile: file of real;

{Plotting variables}
        xAr, fxAr: Plot2array;
        NumxIncPlusOne: integer;
```

```
        width, height, ULx, ULy: integer;
        ClipScaley: YesNo;
        LoClipy, HiClipy: real;
        xTicBeg, xTicInc, xMinTicInc: real;
        yTicBeg, yTicInc, yMinTicInc: real;
        xLabLen, yLabLen: integer;
        xTicType, yTicType: TicType;
        xMinTic: TicType;
        yMinTic: TicType;
        LineWidth: integer; {width of the plotted line}
        xLabel, yLabel: string;
        error: string;

    function CircleTest (xSolution, xCircle, yCircle, aCircle: extended): extended;
{Tests which of the ellipse solutions, e.g., (x,y) or (x,-y), intersects the}
{specified circle.}
        var
            y1, y2, PosCircle, NegCircle: extended;
    begin
{Compute possible y solutions, watching for ill-conditioning and round-off}
{error where 1.0 - sqr(xSolution / AA0) can be slightly negative.}
    y1 := B0 * sqrt(abs(1.0 - sqr(xSolution / AA0)));
    y2 := -y1; {negative square root}
    PosCircle := sqr(xSolution - xCircle) + sqr(y1 - yCircle) - sqr(aCircle);
    NegCircle := sqr(xSolution - xCircle) + sqr(y2 - yCircle) - sqr(aCircle);
    if abs(PosCircle) <= abs(NegCircle) then {One or the other should be zero.}
        CircleTest := y1 {choose positive square root}
    else
        CircleTest := y2; {choose negative square root}
    end; {CircleTest}

    function EAPofTophat (xTH, yTH, aTH: extended): extended;
{Calculates elliptical-arc projections of a top hat.}
{(xTH, yTH) is the center of the top hat, aTH is the radius.}
{The ellipse has its foci on the y-axis always (i.e., bistatic coordinates).}
        var
            xbi, ybi: extended; {top hat coords in bistatic system}
            A1, B1, C1, a, b, c, d, ya, yb, yc, yd, x1, y1, x2, y2: extended;
            theta1, theta2, dummy, arclength1, arclength2, CompleteEllipticIntegral: extended;
            LineIntegral1, LineIntegral2, phi1, phi2: extended;
            HowManyRoots: longint;
            denom: extended;
            FirstRoot: boolean;
            thetaNormal, thetaPosWrap, thetaNegWrap, thetaMin: extended;
            FixEllipticIntegral: boolean;
    begin
{Transform top hat center to bistatic coordinates.}
    xbi := (xTH - bigX0) * costhetab + (yTH - bigY0) * sinthetab;
    ybi := -(xTH - bigX0) * sinthetab + (yTH - bigY0) * costhetab;

{Find intersections of ellipse (or circle) and the top hat circle.}
    B1 := -2.0 * xbi;
    C1 := sqr(xbi) + sqr(B0) + sqr(ybi) - sqr(aTH);
    if not Monostatic then {Bistatic}
        begin
        A1 := 1.0 - sqr(B0 / AA0);
        a := 2.0 * B1 / A1;
        b := 2.0 * C1 / A1 + sqr(B1 / A1) + 4.0 * sqr((ybi * B0) / (AA0 * A1));
        c := 2.0 * B1 * C1 / sqr(A1);
        d := (sqr(C1) - 4.0 * sqr(ybi * B0)) / sqr(A1);
        SolveQuartic(a, b, c, d, HowManyRoots)
```

```
        end
    else {Monostatic}
        begin
        denom := sqr(B1) + 4.0 * sqr(ybi);
        b := 2.0 * B1 * C1 / denom;
        c := (sqr(C1) - sqr(2.0 * ybi * B0)) / denom;
        SolveQuadratic(b, c, HowManyRoots);

{Fix up the "other roots" to be QNaNs, just like they would be in SolveQuartic.}
        a := sqrt(-1.0);
        d := sqrt(-1.0);
        end;

{Find y-solutions of the two real-valued quartic equation solutions.}
    if HowManyRoots = 2 then {Ellipse and circle intersect normally.}
        begin
        FirstRoot := true;
        if ClassExtended(a) <> QNaN then
            begin
            ya := CircleTest(a, xbi, ybi, aTH);
            if FirstRoot then
                begin
                x1 := a;
                y1 := ya;
                FirstRoot := false
                end
            end;
        if ClassExtended(b) <> QNaN then
            begin
            yb := CircleTest(b, xbi, ybi, aTH);
            if FirstRoot then
                begin
                x1 := b;
                y1 := yb;
                FirstRoot := false
                end
            else
                begin
                x2 := b;
                y2 := yb;
                end;
            end;
        if ClassExtended(c) <> QNaN then
            begin
            yc := CircleTest(c, xbi, ybi, aTH);
            if FirstRoot then
                begin
                x1 := c;
                y1 := yc;
                FirstRoot := false
                end
            else
                begin
                x2 := c;
                y2 := yc;
                end;
            end;
        if ClassExtended(d) <> QNaN then
            begin {First root has been found.}
            yd := CircleTest(d, xbi, ybi, aTH);
            x2 := d;
```

```
            y2 := yd
            end;

{begin finding the arc length between (x1, y1) and (x2, y2)}
        theta1 := ArcTan2(y1, x1);
        if (theta1 < 0) then
            theta1 := theta1 + TwoPi; {Want it between 0 and 2π to start with.}
        theta2 := ArcTan2(y2, x2);
        if (theta2 < 0) then
            theta2 := theta2 + TwoPi;

{Make theta2 > theta1.  x1, y1, x2, y2 are left as is because they aren't used after this.}
        if theta1 > theta2 then
            begin
            dummy := theta1;
            theta1 := theta2;
            theta2 := dummy;
            end;

{Check for a condition in finding the elliptical integral.}
        if (theta2 - theta1) > Pi then {intersections in the I and IV bistatic quadrants}
            FixEllipticIntegral := true
        else
            FixEllipticIntegral := false;

        if Weighting = NoWeight then
            begin

{Find the first and second arc lengths.}
            CompleteEllipticIntegral := el2(tan(PiOverTwo), kc, 1.0, kcsquared); {Arc length from 0 to pi/2.}
            arclength2 := el2(AA0 * tan(theta2) / B0, kc, 1.0, kcsquared) + (2.0 * trunc((theta2 + PiOverTwo) / pi)) *
CompleteEllipticIntegral;
            arclength1 := el2(AA0 * tan(theta1) / B0, kc, 1.0, kcsquared) + (2.0 * trunc((theta1 + PiOverTwo) / pi)) *
CompleteEllipticIntegral;

            if not FixEllipticIntegral then
                EAPofTophat := B0 * (arclength2 - arclength1)
            else
                EAPofTopHat := B0 * (4.0 * CompleteEllipticIntegral - (arclength2 - arclength1));
            end
        else if Weighting = Attenuation then
            begin

{Find the first and second line integrals.}
            CompleteEllipticIntegral := el2(tan(PiOverTwo), kc, 1.0, 1.0); {Line integral from 0 to pi/2.}
            LineIntegral2 := el2(AA0 * tan(theta2) / B0, kc, 1.0, 1.0) + (2.0 * trunc((theta2 + PiOverTwo) / pi)) *
CompleteEllipticIntegral;
            LineIntegral1 := el2(AA0 * tan(theta1) / B0, kc, 1.0, 1.0) + (2.0 * trunc((theta1 + PiOverTwo) / pi)) *
CompleteEllipticIntegral;
            if not FixEllipticIntegral then
                EAPofTophat := (LineIntegral2 - LineIntegral1) / B0
            else
                EAPofTopHat := (4.0 * CompleteEllipticIntegral - (LineIntegral2 - LineIntegral1)) / B0;
            end;
        end {HowManyRoots = 2}
    else if HowManyRoots = 0 then
        EAPofTophat := 0.0 {No intersection of ellipse and top hat.}
    else
        begin
        write('Quartic found ', HowManyRoots : 1, 'roots.  It should have found 0 or 2.');
        SysBeep(1);
```

```
        SysBeep(1);
        SysBeep(1);
        SysBeep(1);
        SysBeep(1);
        readln;
        EAPofTophat := 0.0; {Don't know what else to do here.}
        end
    end; {EAPofTopHat}

    function fx (B0: extended): extended;
{Finds the number of roots minus one in the solution of the circlular ground}
{patch and an ellipse.  Used in the bisection solver rtbis to find the values of}
{B0 which cause tangency, so that the sample rate in B0 can be set.}
        var
            HowManyRoots: longint;
            x0, y0, aaa0, A0, A1, B1, C1, a, b, c, d: extended;
            r1, r2, r3, r4: extended;
    begin
{Transform ground patch center to bistatic coordinates.}
    x0 := -bigX0 * costhetab - bigY0 * sinthetab;
    y0 := bigX0 * sinthetab - bigY0 * costhetab;

{Set radius of the ground patch, in pixels.}
    aaa0 := 63.0;

{Compute some coefficients.}
    A0 := sqrt(sqr(B0) - sqr(F)); {x-axis intercept of the ellipse}
    A1 := 1.0 - sqr(B0) / sqr(A0);
    B1 := -2.0 * x0;
    C1 := sqr(x0) + sqr(B0) + sqr(y0) - sqr(aaa0);
    a := 2.0 * B1 / A1;
    b := 2.0 * C1 / A1 + sqr(B1 / A1) + 4.0 * sqr(y0) * sqr(B0) / sqr(A0 * A1);
    c := 2.0 * B1 * C1 / sqr(A1);
    d := (sqr(C1) - 4.0 * sqr(y0 * B0)) / sqr(A1);
    SolveQuartic(a, b, c, d, HowManyRoots);
    fx := HowManyRoots - 1;
    end; {fx}

begin {main}

{Generate quiet NaNs on sqrt(-1.0), for counting real roots in polynomial solvers.}
SetHalt(Invalid, false);

{Set up text window.}
SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Get weighting information.}
write('Type of Weighting [NoWeight, Attenuation]: ');
readln(Weighting);

{Prepare output file.}
FileName := NewFileName('File for projections:');
if FileName <> '' then
    rewrite(outFile, FileName)
else
    ExitToShell;
write(outFile, Nrays); {aka npx}
write(outFile, Nproj); {aka npy}
```

```
TrajectoryFileName := NewFileName('File for trajectories:');
if TrajectoryFileName <> '' then
    rewrite(TrajectoryFile, TrajectoryFileName)
else
    ExitToShell;

{Initialize the unused slot in the projection array.}
proj[-64] := 0.0;

{Get initial transmitter and receiver coordinates.}
writeln('Vertical:   Transmitter at (72, -50), Receiver at (72, 50)');
writeln('Horizontal:  Transmitter at (172, 0), Receiver at (72, 0)');
writeln('Oblique:   Transmitter at (144.61955, 44.21463), Receiver at (68.85394, -21.05076)');
write('Enter initial bistatic orientation:  ');
readln(InitialPosition);

{Set parameters according to initial coordinates.}
case InitialPosition of
Vertical:
    begin
    Xt0 := 72.0;
    Yt0 := -50.0;
    Xr0 := 72.0;
    Yr0 := 50.0;
    B0minLast := 50.8035;
    B0maxLast := 143.9618;
    end;
Horizontal:
    begin
    if (OrbModFreq = 0.0) and (OrbModDepth = 0.0) then
        begin
        Xt0 := 172.0;
        Yt0 := 0.0;
        Xr0 := 72.0;
        Yr0 := 0.0;
        B0minLast := 59.0;
        B0maxLast := 185.0
        end
    else
        begin

{Need a little asymmetry to avoid a small error at 180°.}
        Xt0 := 171.999887209;
        Yt0 := 9.99999456912e-2;
        Xr0 := 71.9999878313;
        Yr0 := -0.041860462758;
        B0minLast := 59.0000029;
        B0maxLast := 184.999952;
        end;
    end;
Oblique:
    begin
    Xt0 := 144.61955;
    Yt0 := 44.21463;
    Xr0 := 68.85394;
    Yr0 := -21.05076;
    B0minLast := 57.5891748;
    B0maxLast := 173.092796;
    end;
end;
```

```
{Find the initial focal length of the bistatic system.  This will be modulated later.}
F0 := 0.5 * sqrt(sqr(Xt0 - Xr0) + sqr(Yt0 - Yr0)); {evaluates to 50 for Vertical, Horizontal, Oblique cases.}

{Find distance to center of bistatic system.}
initialR0 := 0.5 * sqrt(sqr(Xt0 + Xr0) + sqr(Yt0 + Yr0));

{Get shutdown info.}
write('Do you want to shut down when done? [yes/no] ');
readln(WantShutDown);

{Find initial tilt angle of the bistatic axes.}
if (Xt0 = Xr0) and (Yt0 = Yr0) then {Monostatic SAR}
    begin
    Monostatic := true;
    writeln('Monostatic radar.');
    thetab0 := 0.0 {Actually, thetab0 doesn't make sense in this case.}
    end
else
    begin
    thetab0 := ArcTan2(Xt0 - Xr0, -(Yt0 - Yr0));
    if (thetab0 = 0.0) or (thetab0 = pi) then
        thetab0 := thetab0 + 1.0e-4; {SolveQuartic doesn't like 0.0 or pi.}
    end;

{Find the initial polar coordinates of the transmitter and receiver.}
Rt0 := sqrt(sqr(Xt0) + sqr(Yt0));
Thetat0 := ArcTan2(Yt0, Xt0);
Rr0 := sqrt(sqr(Xr0) + sqr(Yr0));
Thetar0 := ArcTan2(Yr0, Xr0);
if abs(Thetat0 + Thetar0) > 0.0001 then
    begin
    writeln('Warning:  The initial radar angle is not zero.');
    readln;
    ExitToShell
    end;

{Set coordinates and radii of the top hats.}
r0 := 20.0; {0.  Distance of center of top hat from origin}
theta0 := 135.9 * RadiansPerDegree; {Angle of center of top hat from origin}
x0 := r0 * cos(theta0); {x-coordinate of center.}
y0 := r0 * sin(theta0); {y-coordinate of center.}
a0 := 35.0; {Radius of the top hat}
h0 := 0.95; {Height of top hat}

r1 := 60.0; {1}
theta1 := 47.0 * RadiansPerDegree;
x1 := r1 * cos(theta1);
y1 := r1 * sin(theta1);
a1 := 2.0;
h1 := 0.75;

r2 := 10.0; {2}
theta2 := 135.9 * RadiansPerDegree;
x2 := r2 * cos(theta2);
y2 := r2 * sin(theta2);
a2 := 20.0;
h2 := -0.2;

r3 := 40.0; {3}
theta3 := -89.1 * RadiansPerDegree;
x3 := r3 * cos(theta3);
```

```
y3 := r3 * sin(theta3);
a3 := 5.0;
h3 := 0.5;

{Compute elliptical-arc-projection transform of the top hat function.}
{begin loop over angle--Nproj projections between 0 and 2*pi.}
AngleInc := TwoPi / Nproj;
for i := 0 to Nprojm1 do
    begin
    writeln(i : 1); {Monitor progress.}

    Angle := i * AngleInc; {Angle of the whole bistatic system, not counting initial tilt.}
    F := F0 * (1.0 - OrbModDepth + OrbModDepth * cos(OrbModFreq * Angle));

{Calculate trajectories of transmitter and receiver.}
    Xt[i] := initialR0 * cos(Angle) + F * sin(RotateRate * Angle + thetab0);
    Yt[i] := initialR0 * sin(Angle) - F * cos(RotateRate * Angle + thetab0);

    Xr[i] := initialR0 * cos(Angle) - F * sin(RotateRate * Angle + thetab0);
    Yr[i] := initialR0 * sin(Angle) + F * cos(RotateRate * Angle + thetab0);

{Calculate the center of the bistatic coordinate system}
    bigX0 := 0.5 * (Xt[i] + Xr[i]);
    bigY0 := 0.5 * (Yt[i] + Yr[i]);

{Calculate the tilt angle of the bistatic coordinate system.}
    thetab := ArcTan2(Xt[i] - Xr[i], -(Yt[i] - Yr[i]));
    costhetab := cos(thetab);
    sinthetab := sin(thetab);

{Find B0min.}
    B0minLo := max(0.9 * B0minLast, F * (1.0000001)); {Bracket the nearer root; don't go below F.}
    B0minHi := 1.1 * B0minLast;
    B0min[i] := rtbis(B0minLo, B0minHi, small);
    B0minLast := B0min[i]; {Should be about the same next time.}
    B0min[i] := B0min[i] + 2.0e-8; {Guarantee at least one real root.}

{Find B0max.}
    B0maxLo := 0.9 * B0maxLast; {Bracket the farther root.}
    B0maxHi := 1.1 * B0maxLast;
    B0max[i] := rtbis(B0maxLo, B0maxHi, small);
    B0maxLast := B0max[i]; {Should be about the same next time.}
    B0max[i] := B0max[i] - 2.0e-8; {Guarantee at least one real root.}

    B0scale := (B0max[i] - B0min[i]) / (Nrays - 2.0);

{Compute each projection.}
    for j := -(Nrays div 2 - 1) to Nrays div 2 - 1 do
        begin
        B0 := B0min[i] + ((Nrays - 2.0) / 2.0 - j) * B0scale; {y-axis intercept of the ellipse.}
        AA0 := sqrt(sqr(B0) - sqr(F)); {x-axis intercept of the ellipse}
        ksquared := 1 - sqr(AA0 / B0);
        kcsquared := 1 - ksquared;
        kc := sqrt(kcsquared);
        proj[j] := h0 * EAPofTophat(x0, y0, a0) + h1 * EAPofTophat(x1, y1, a1) + h2 * EAPofTophat(x2, y2, a2) + h3 *
EAPofTophat(x3, y3, a3);
        end;

{Write each projection to a file.}
    for j := -Nrays div 2 to Nrays div 2 - 1 do
        write(outFile, proj[j]);
```

```
{Get a chance to quit.}
    if button then
        Halt;
    end;

{Write trajectory information to a file, to be read by reconstruction program.}
for i := 0 to Nprojm1 do
    write(TrajectoryFile, Xt[i]);
for i := 0 to Nprojm1 do
    write(TrajectoryFile, Yt[i]);
for i := 0 to Nprojm1 do
    write(TrajectoryFile, Xr[i]);
for i := 0 to Nprojm1 do
    write(TrajectoryFile, Yr[i]);
for i := 0 to Nprojm1 do
    write(TrajectoryFile, B0min[i]);
for i := 0 to Nprojm1 do
    write(TrajectoryFile, B0max[i]);

{Plot the trajectories.}
for i := 0 to Nprojm1 do
    begin
    xPlot[i + 1] := Xt[i]; {transmitter}
    yPlot[i + 1] := Yt[i];

    xPlot[i + Nproj + 1] := Xr[i]; {receiver}
    yPlot[i + Nproj + 1] := Yr[i];
    end;

{Set plotting paramters.}
width := 385;
height := 385;
ULx := 2;
ULy := 12;
ClipScaley := no;
LoClipy := 0.0; {Don't care}
HiClipy := 0.0; {Don't care}
xLabel := 'none';
xLabLen := 4;
yLabel := 'none';
yLabLen := 4;
xTicType := none;
xTicBeg := 0.0; {Don't care}
xTicInc := 0.0; {Don't care}
xMinTic := none;
xMinTicInc := 0.0; { Don't care}
yTicType := none;
yTicBeg := 0.0; {Don't care}
yTicInc := 0.0; {Don't care}
yMinTic := none;
yMinTicInc := 0.0; {Don't care}
LineWidth := 1;

{Plot it.}
SetRect(DrawingRect, 2, 40, 417, 477);
SetDrawingRect(DrawingRect);
ShowDrawing;

SysBeep(1);
SysBeep(1);
```

```
SysBeep(1);
SysBeep(1);
SysBeep(1);

Plotter2(xPlot, yPlot, 2 * Nproj, width, height, ULx, ULy, Linear, ClipScaley, LoClipy, HiClipy, xLabel, xLabLen, yLabel,
yLabLen, xTicType, xTicBeg, xTicInc, xMinTic, xMinTicInc, yTicType, yTicBeg, yTicInc, yMinTic, yMinTicInc, LineWidth,
Black, error);
if WantShutDown = no then
    repeat
    until button;

{Print out exit info to verify flight trajectory selection, etc.}
ShowText;
writeln('Elliptical-arc projections.');
writeln('Weighting: ', Weighting);
writeln('Output file name is ', FileName, '.');

if (WantShutDown = yes) and (not Button) then
    ShutDwnPower
else
    pause('Done');
end.
```

```
program EACBPU (input, output);
{This program does Elliptical-Arc Convolution BackProjection from Unattenuated}
{projections and generalized transmitter and receiver trajectories.}
    uses
        SANE, RealFunctions, ComplexFunctions, FFT256, ShutDownManager;
    label
        99;
{Dimension-dependent things.}
    const
        RaysPerProj = 128;
        RaysPerProjMinusOne = RaysPerProj - 1;
        RaysPerProjTimesTwo = RaysPerProj * 2;
        NumFFTPoints = RaysPerProjTimesTwo;
        ImSize = 128;  {Square image}
        ImSizeOv2 = ImSize div 2;
        ImSizeOv2Min1 = ImSize div 2 - 1;
{end of dimension-dependent things.}
    type
    Row = array[-ImSizeOv2..ImSizeOv2] of real;                    {A}
        RowPtr = ^Row;                                              {large}
        ImageArray = array[-ImSizeOv2..ImSizeOv2] of RowPtr; {array}
        AnArray = array[0..197] of extended;
        ptrAnArray = ^AnArray;
        YesNo = (yes, no);
    var
        Xt, Yt: ptrAnArray; {coords of transmitter}
        Xr, Yr: ptrAnArray; {coords of receiver}
        B0min, B0max: ptrAnArray; {begin and end of sampling}
        Proj: array [1..RaysPerProj] of extended; {Projection (128)}
        fProj: array[-RaysPerProj..RaysPerProjMinusOne] of extended; {Filtered projection (-128:127)}
        image: ImageArray; {Indexed as image[row]^[col] — this is the reconstructed image}
        R0ar: array[0..199] of extended; {array for variable projector radius.}
        theta0ar: array[0..199] of extended;
        Xt0, Yt0, Xr0, Yr0, thetab0, Rt0, Thetat0, Rr0, Thetar0: extended;
        AngleInc, Angle, bigX0, bigY0, thetab, costhetab, sinthetab: extended;
```

```
        xb, xbsquared, yb, AA0, weight, F, B0, B0mid, B0scale, B0weight: extended; {elliptic parameters}
        OverallWeight: extended; {relative weight of B0min, per projection, relative to B0max}
        C0, D0: extended; {hyperbolic parameters}
        h: FFT256array;  {filter impulse response (256)}
        clProj: FFT256array;  {Zero-padded complex version of Proj}
        k, i, j, npx, npy, mm, it, npxo2: longint;
        BegTime, FinTime: longint;
        Nproj, Nprojm1, Nprojo2m1: longint;
        PixelSkip, PixelCount: longint; {for low-resolution plots.}
        rnpx, rnpy, costheta, sintheta, xcostheta, ysintheta, jj, rit: extended;
        xsintheta, ycostheta, R0: extended;
        ymbigY0, ymbigY0sinthetab, ymbigY0costhetab: extended;
        x, y, xp, yp, xpp, rho, littletheta: extended; {Coordinate transformation stuff}
        X0, Y0, InitX0, vT, AA, SS: extended;
        WantWindow, quit, UpdateFile, WantShutDown: YesNo;
        inFileName, outFileName, TrajectoryFileName: string;
        inFile, outFile, TrajectoryFile: file of real;
        TextRect: rect;
        Fsquared, distance1, distance2: extended;

    procedure Pause (PauseMessage: string);
    begin
    write(PauseMessage);
    readln;
    end;

    procedure InitArray (var TheArray: ImageArray;
                                    NumRows: longint;
                                    NumCols: longint);
    {Allocates memory from the heap for a large array.}
        var
            i: longint;
    begin
    for i := -NumRows div 2 to NumRows div 2 do
        TheArray[i] := RowPtr(NewPtr(SizeOf(real) * (NumCols + 1)));
    end;

begin {EACBP}
{Set IEEE halt conditions.}
SetHalt(Invalid, true);
SetHalt(Underflow, false);
SetHalt(Overflow, false);
SetHalt(DivByZero, false);
SetHalt(Inexact, false);

{Allot some memory.}
New(Xt);
New(Yt);
New(Xr);
New(Yr);
New(B0min);
New(B0max);
InitArray(image, ImSize, ImSize);

SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Establish file connections.}
inFileName := OldFileName('Projection file:');
if inFileName <> '' then
```

```
    reset(inFile, inFileName)
else
    ExitToShell;

outFileName := NewFileName('File for reconstruction:');
if outFileName <> '' then
    rewrite(outFile, outFileName)
else
    ExitToShell;

TrajectoryFileName := OldFileName('File for trajectories:');
if TrajectoryFileName <> '' then
    reset(TrajectoryFile, TrajectoryFileName)
else
    ExitToShell;

{Set overall weight, per projection.}
writeln('Overall weight of B0min relative to B0max, per projection (suggest 2.0): ');
readln(OverallWeight);

{Select image resolution parameter.}
write('Enter N, for skipping every Nth image pixel: ');
readln(PixelSkip);
if PixelSkip < 1 then
    begin
    pause('N must be >= 1.  Quitting.');
    ExitToShell
    end;

{Get data sizes.}
read(inFile, rnpx); {Number of rays per projection}
read(inFile, rnpy); {Number of projections}
npx := round(rnpx);
npy := round(rnpy);
npxo2 := npx div 2;
Nproj := npy;
Nprojm1 := Nproj - 1;
Nprojo2m1 := Nproj div 2 - 1;

{Read in arrays of transmitter and receiver trajectories—should match data collection case.}
{Also read in values of B0min and B0max for each projection.}
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Xt^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Yt^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Xr^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Yr^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, B0min^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, B0max^[i]);

{Set up windowing option--Hamming or none.}
write('Do you want a smoothing window? [yes/no] ');
readln(WantWindow);

{Optional shutdown when program is done.}
write('Do you want to shut down when the program is done? [yes/no] ');
readln(WantShutDown);
```

```
SysBeep(1);
GetDateTime(BegTime);

mm := 1 + round(log10(rnpx) / log10(2.0)); {FFT "size"}

{Prepare frequency response array, for later filtering--see Haykin, p. 392.}
h[npxo2 + 1] := cmplx(0.125, 0.0); {n = npx/2 (64)}
j := npxo2 + 3;
while j <= npx - 1 do { n:=66..126, even}
    begin
    h[j] := cmplx(0.0, 0.0);
    h[npx + 2 - j] := cmplx(0.0, 0.0); {n:=62..2, even}
    j := j + 2
    end;
h[1] := cmplx(0.0, 0.0); {n:=0}
j := npxo2 + 2;
while j <= npx do {n:=65..127, odd}
    begin
    jj := j - npxo2 - 1;
    h[j] := cmplx(-0.5 / (sqr(jj) * sqr(Pi)), 0.0);
    h[npx + 2 - j] := h[j]; {n:=63..1, n odd}
    j := j + 2
    end;

{Zero pad the impulse response array.}
for j := npx + 1 to 2 * npx do
    h[j] := cmplx(0.0, 0.0);

FFT256(h, 1.0);

{Apply the optional smoothing window (Hamming).}
if WantWindow = yes then
    for j := 1 to 2 * npx do
        h[j] := c(h[j], m, cmplx((0.54 - 0.46 * cos(2.0 * Pi * (j - 1 - npx) / (2.0 * npx))), 0.0));

{Initialize the reconstructed-image array—code outliers with -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if sqrt(i * i + j * j) <= ImSizeOv2Min1 then
            image[i]^[j] := 0.0
        else
            image[i]^[j] := -INF;

{Begin loop over all filtered projections.}
for k := 0 to Nprojm1 do {First one is the projection from angle 0.}
    begin

    writeln('Projection number ', k : 1); {Monitor progress}

{Some constants for indexing the filtered projection data.}
    B0mid := (B0min^[k] + B0max^[k]) / 2.0;
    B0scale := 2.0 * ImSizeOv2Min1 / (B0max^[k] - B0min^[k]);

{Overall weighting, per projection.}
    B0weight := B0max^[k] - OverallWeight * B0min^[k];

 {Center of bistatic coordinate system}
    bigX0 := 0.5 * (Xt^[k] + Xr^[k]);
    bigY0 := 0.5 * (Yt^[k] + Yr^[k]);
```

```
{Set the focal distance of the ellipse.}
    F := 0.5 * sqrt(sqr(Xt^[k] - Xr^[k]) + sqr(Yt^[k] - Yr^[k]));
    Fsquared := sqr(F);

 {Find the tilt angle of bistatic coordinate system.}
    thetab := ArcTan2(Xt^[k] - Xr^[k], -(Yt^[k] - Yr^[k]));
    costhetab := cos(thetab);
    sinthetab := sin(thetab);

{Get a projection from the input file}
    for j := 1 to npx do
        read(inFile, Proj[j]);

{Apply the overall weighting to the current projection.}
    for j := 1 to npx do
        Proj[j] := Proj[j] * B0weight;

    {Make zero-padded, complex version of the projection.}
    for j := 1 to npx do
        clProj[j] := cmplx(Proj[j], 0.0);
    for j := npx + 1 to 2 * npx do {zero padding}
        clProj[j] := cmplx(0.0, 0.0);

    {Filter the projections.}
    FFT256(clProj, 1.0);
    for j := 1 to 2 * npx do
        clProj[j] := c(clProj[j], m, h[j]);
    FFT256(clProj, -1.0);

    {Convert the double-length filtered projection to real form,}
    {and change indexing to image coordinates.}
    for j := 1 to 2 * npx do
        fProj[j - npx - 1] := clProj[j].re;

    {Double loop over all pixels.}
    i := -ImSizeOv2;
    while i <= ImSizeOv2 do
        begin
        y := i; {y:=float(i)}
        ymbigY0 := (y - bigY0);
        ymbigY0sinthetab := ymbigY0 * sinthetab;
        ymbigY0costhetab := ymbigY0 * costhetab;

        j := -ImSizeOv2;
        while j <= ImSizeOv2 do
            begin
            if image[i]^[j] <> -INF then {not outside the reconstruction circle.}
                begin
                x := j; {x:=float(j)}

{Transform pixel locations to bistatic coordinates.}
                xb := (x - bigX0) * costhetab + ymbigY0sinthetab;
                yb := (bigX0 - x) * sinthetab + ymbigY0costhetab;
                xbsquared := sqr(xb);

                distance1 := sqrt(sqr(xb) + sqr(F + yb));
                distance2 := sqrt(sqr(xb) + sqr(F - yb));
                B0 := 0.5 * (distance1 + distance2);
                C0 := 0.5 * (distance1 - distance2);
                AA0 := sqrt(sqr(B0) - Fsquared); {x-axis intercept of the ellipse}
                D0 := sqrt(abs(Fsquared - sqr(C0))); {Remove abs() unless "172, 72" case.}
```

```
                xpp := B0scale * (B0mid - B0);

 {Index into fProj, to begin interpolation—round towards minus infinity.}
                it := round(xpp - 0.5); {index for linear interpolation}
                rit := it;

{Find weight, for doing a weighted backprojection.}
                weight := sqr(AA0 * F) / (sqr(AA0 * C0) + sqr(B0 * D0));

{Form image using linear interpolation.}
                image[i]^[j] := image[i]^[j] + weight * (fProj[it] + (xpp - rit) * (fProj[it + 1] - fProj[it]));
                end;
            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [yes/no] ');
            readln(quit);
            if quit = yes then
                begin
                write('Do you want to update the output file? [yes/no] ');
                readln(UpdateFile);
                if UpdateFile = yes then
                    goto 99
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

99:
{Fix the outlying pixels which were set to -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if image[i]^[j] = -INF then
            image[i]^[j] := 0.0;

{Prepare to write image to output file.}
close(inFile);

{Begin by finding how many non-zero pixels, in case of low-res.}
PixelCount := 0;
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    PixelCount := PixelCount + 1;
    i := i + PixelSkip
    end;
write(outFile, PixelCount); {Actual image size, in pixels.}
write(outFile, PixelCount); {Square image}

{Write the image to the output file.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    j := -ImSizeOv2;
    while j <= ImSizeOv2 do
        begin
```

```
            write(outFile, image[i]^[j]); {Now write it.}
            j := j + PixelSkip
            end;
        i := i + PixelSkip
        end;
close(outFile);

{Print out exit info.}
if WantWindow = yes then
    writeln('Hamming window');

GetDateTime(FinTime);
writeln('Execution time was: ', (FinTime - BegTime) / 60.0 : 5 : 1, ' minutes.');
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);

writeln('Input file: ', inFileName);
writeln('Output file: ', outFileName);
writeln('Trajectory file: ', TrajectoryFileName);
writeln('Overall weight: ', OverallWeight : 13 : 3);

if (WantShutDown = yes) and (not Button) then
    ShutDwnPower {Make sure all updated files are closed first.}
else
    pause('Done');
end.
```

```
program EACBPA (input, output);
{This program does Elliptical-Arc Convolution BackProjection from Attenuated}
{projections and generalized transmitter and receiver trajectories.}
    uses
        SANE, RealFunctions, ComplexFunctions, FFT256, ShutDownManager;
    label
        99;
{Dimension-dependent things.}
    const
        RaysPerProj = 128;
        RaysPerProjMinusOne = RaysPerProj - 1;
        RaysPerProjTimesTwo = RaysPerProj * 2;
        RaysPerProjOv2 = RaysPerProj div 2;
        RaysPerProjOv2Min1 = RaysPerProjOv2 - 1;
        RaysPerProjOv2Plus1 = RaysPerProjOv2 + 1;
        NumFFTPoints = RaysPerProjTimesTwo;
        ImSize = 128; {Square image size}
        ImSizeOv2 = ImSize div 2;
        ImSizeOv2Min1 = ImSize div 2 - 1;
{end of dimension-dependent things.}
    type
    Row = array[-ImSizeOv2..ImSizeOv2] of real;                    {A}
        RowPtr = ^Row;                                              {large}
        ImageArray = array[-ImSizeOv2..ImSizeOv2] of RowPtr; {array}
        AnArray = array[0..199] of extended;
        ptrAnArray = ^AnArray;
        YesNo = (yes, no);
    var
```

```
        Xt, Yt: ptrAnArray; {coords of transmitter}
        Xr, Yr: ptrAnArray; {coords of receiver}
        B0min, B0max: ptrAnArray; {begin and end of sampling}
        rh: array[1..RaysPerProj] of extended; {Real version of impulse response}
        Proj: array[-RaysPerProjOv2..RaysPerProjOv2Min1] of real; {projection}
        wProj: array[-RaysPerProjOv2..RaysPerProjOv2Min1] of extended; {weighted projection}
        FcoshVarray, FsinhVarray: array[-RaysPerProjOv2..RaysPerProjOv2Min1] of extended;
        image: ImageArray; {Indexed as image[row]^[col] — this is the reconstructed image}
        Xt0, Yt0, Xr0, Yr0, thetab0, Rt0, Thetat0, Rr0, Thetar0: extended;
        AngleInc, Angle, bigX0, bigY0, thetab, costhetab, sinthetab: extended;
        xb, yb, xbsquared, AA0, weight, F, B0, B0mid, B0scale, OneOvB0scale: extended; {elliptic parameters}
        B0weight: extended;
        OverallWeight: extended; {relative weight of B0min, per projection, relative to B0max}
        distance1, distance2, Fsquared: extended;
        C0, D0, B0dum, sinU, cosU, sinhV, coshV, xw, yw, xwsquared, sum1, sum2: extended;
        h: FFT256array;  {filter impulse response (256)}
        clProj: FFT256array;  {Zero-padded complex version of Proj}
        k, i, j, npx, npy, mm, it, iit, t, index, itplus1, npxo2: longint;
        l, lPlusImSizeOv2, n: longint;
        BegTime, FinTime: longint;
        Nproj, Nprojm1, Nprojo2m1: longint;
        PixelSkip, PixelCount: longint; {for low-resolution plots.}
        rnpx, rnpy, costheta, sintheta, xcostheta, ysintheta, jj, rit: extended;
        xsintheta, ycostheta, R0: extended;
        ymbigY0, ymbigY0sinthetab, ymbigY0costhetab: extended;
        x, y, xp, yp, xpp, rho, littletheta: extended; {Coordinate transformation stuff}
        X0, Y0, InitX0, vT, AA, SS: extended;
        WantWindow, quit, UpdateFile, WantShutDown: YesNo;
        inFileName, outFileName, TrajectoryFileName: string;
        inFile, outFile, TrajectoryFile: file of real;
        TextRect: rect;

        TimeToRunFile : text;

    procedure Pause (PauseMessage: string);
    begin
    write(PauseMessage);
    readln;
    end;

    procedure InitArray (var TheArray: ImageArray;
                                    NumRows: longint;
                                    NumCols: longint);
    {Allocates memory from the heap for a large array.}
        var
            i: longint;
    begin
    for i := -NumRows div 2 to NumRows div 2 do
        TheArray[i] := RowPtr(NewPtr(SizeOf(real) * (NumCols + 1)));
    end;

begin{EACBP}

{Set behavior of IEEE floating point.}
SetHalt(Invalid, true);

{Allot some memory.}
InitArray(image, ImSize, ImSize);
New(Xt);
New(Yt);
New(Xr);
```

```
New(Yr);
New(B0min);
New(B0max);

SetRect(TextRect, 2, 40, 637, 477);
SetTextRect(TextRect);
ShowText;

{Establish file connections.}
inFileName := OldFileName('Projection file:');
if inFileName <> '' then
    reset(inFile, inFileName)
else
    ExitToShell;

outFileName := NewFileName('File for reconstruction:');
if outFileName <> '' then
    rewrite(outFile, outFileName)
else
    ExitToShell;

TrajectoryFileName := OldFileName('File for trajectories:');
if TrajectoryFileName <> '' then
    reset(TrajectoryFile, TrajectoryFileName)
else
    ExitToShell;

{Set overall weight, per projection.}
writeln('Overall weight of B0min relative to B0max, per projection: ');
readln(OverallWeight);

{Select image resolution parameter.}
write('Enter N, for skipping every Nth image pixel: ');
readln(PixelSkip);
if PixelSkip < 1 then
    begin
    pause('N must be >= 1.  Quitting.');
    ExitToShell
    end;

{Get data sizes.}
read(inFile, rnpx); {Number of rays per projection}
read(inFile, rnpy); {Number of projections}
npx := round(rnpx);
npy := round(rnpy);
npxo2 := npx div 2;
Nproj := npy;
Nprojm1 := Nproj - 1;
Nprojo2m1 := Nproj div 2 - 1;

{Read in arrays of transmitter and receiver trajectories—should match data collection case.}
{Also read in values of B0min and B0max for each projection.}
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Xt^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Yt^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Xr^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, Yr^[i]);
for i := 0 to Nprojm1 do
```

```
    read(TrajectoryFile, B0min^[i]);
for i := 0 to Nprojm1 do
    read(TrajectoryFile, B0max^[i]);

{Set up windowing option--Hamming or none.}
write('Do you want a smoothing window? [yes/no] ');
readln(WantWindow);

{Optional shutdown when program is done.}
write('Do you want to shut down when the program is done? [yes/no] ');
readln(WantShutDown);

SysBeep(1);
GetDateTime(BegTime);

mm := 1 + round(log10(rnpx) / log10(2.0)); {FFT "size"}

{Prepare frequency response array, for later filtering--see Haykin, p. 392.}
h[npxo2 + 1] := cmplx(0.125, 0.0); {n = npx/2 (64)}
j := npxo2 + 3;
while j <= npx - 1 do { n:=66..126, even}
    begin
    h[j] := cmplx(0.0, 0.0);
    h[npx + 2 - j] := cmplx(0.0, 0.0); {n:=62..2, even}
    j := j + 2
    end;
h[1] := cmplx(0.0, 0.0); {n:=0}
j := npxo2 + 2;
while j <= npx do {n:=65..127, odd}
    begin
    jj := j - npxo2 - 1;
    h[j] := cmplx(-0.5 / (sqr(jj) * sqr(Pi)), 0.0);
    h[npx + 2 - j] := h[j]; {n:=63..1, n odd}
    j := j + 2
    end;

{Zero pad the impulse response array.}
for j := npx + 1 to 2 * npx do
    h[j] := cmplx(0.0, 0.0);

{Apply the optional smoothing window (Hamming).}
if WantWindow = yes then
    begin
    FFT256(h, 1.0);
    for j := 1 to 2 * npx do
        h[j] := c(h[j], m, cmplx((0.54 - 0.46 * cos(2.0 * Pi * (j - 1 - npx) / (2.0 * npx))), 0.0));
    FFT256(h, -1.0);
    end;

{Make a real version of h.}
for j := 1 to npx do
    rh[j] := h[j].re;

{Initialize the reconstructed-image array—code outliers with -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if sqrt(i * i + j * j) <= ImSizeOv2Min1 then
            image[i]^[j] := 0.0
        else
            image[i]^[j] := -INF;
```

```
{Begin loop over all filtered projections.}
for k := 0 to Nprojm1 do {First one is the projection from angle 0.}
    begin
    writeln('Projection number ', k : 1); {Monitor progress}

{Some constants for indexing the filtered projection data.}
    B0mid := (B0min^[k] + B0max^[k]) / 2.0;
    B0scale := 2.0 * ImSizeOv2Min1 / (B0max^[k] - B0min^[k]);
    OneOvB0scale := 1.0 / B0scale;

{Overall weighting, per projection.}
    B0weight := B0max^[k] - OverallWeight * B0min^[k];

 {Center of bistatic coordinate system}
    bigX0 := 0.5 * (Xt^[k] + Xr^[k]);
    bigY0 := 0.5 * (Yt^[k] + Yr^[k]);

{Set the focal distance of the ellipse.}
    F := 0.5 * sqrt(sqr(Xt^[k] - Xr^[k]) + sqr(Yt^[k] - Yr^[k]));
    Fsquared := sqr(F);

{Calculate some ellipses for later use in weighting for propagation attenuation.}
    for l := -RaysPerProjOv2Min1 to RaysPerProjOv2Min1 do
        begin
        B0dum := B0min^[k] + (63.0 - l) * OneOvB0scale;
        FcoshVarray[l] := B0dum;
        FsinhVarray[l] := sqrt(sqr(FcoshVarray[l]) - sqr(F));
        end;

 {Find the tilt angle of bistatic coordinate system.}
    thetab := ArcTan2(Xt^[k] - Xr^[k], -(Yt^[k] - Yr^[k]));
    costhetab := cos(thetab);
    sinthetab := sin(thetab);

{Get a projection from the input file}
    for j := -RaysPerProjOv2 to RaysPerProjOv2Min1 do
        read(inFile, Proj[j]);

{Apply the overall weighting to the current projection.}
    for j := -RaysPerProjOv2 to RaysPerProjOv2Min1 do
        Proj[j] := Proj[j] * B0weight;

{Double loop over all pixels.}
    i := -ImSizeOv2;
    while i <= ImSizeOv2 do
        begin
        y := i; {y:=float(i)}
        ymbigY0 := (y - bigY0);
        ymbigY0sinthetab := ymbigY0 * sinthetab; {for transformation to bistatic coords.}
        ymbigY0costhetab := ymbigY0 * costhetab;

        j := -ImSizeOv2;
        while j <= ImSizeOv2 do
            begin
            if image[i]^[j] <> -INF then {not outside the reconstruction circle.}
                begin
                x := j; {x:=float(j)}

{Transform pixel location to bistatic coordinates.}
                xb := (x - bigX0) * costhetab + ymbigY0sinthetab;
                yb := (bigX0 - x) * sinthetab + ymbigY0costhetab;
```

```
                xbsquared := sqr(xb);

{Calculate variables related to the hyperbola.}
                C0 := 0.5 * (sqrt(xbsquared + sqr(yb + F)) - sqrt(xbsquared + sqr(yb - F)));
                sinU := C0 / F;
                cosU := sqrt(1.0 - sqr(sinU));

{Weight the projection with inverse attenuation along a hyperbola through the point (xb, yb).}
                for l := -RaysPerProjOv2Min1 to RaysPerProjOv2Min1 do
                    begin
                    xw := cosU * FsinhVarray[l]; {(xw, yx) are points along the hyperbola through}
                    yw := sinU * FcoshVarray[l]; {the pixel at bistatic coordinate (xb, yb).}
                    xwsquared := sqr(xw);
                    wProj[l] := Proj[l] * sqrt((xwsquared + sqr(F + yw)) * (xwsquared + sqr(F - yw)));
                    end;

{Compute variables needed for weighting along the backprojection.}
                distance1 := sqrt(xbsquared + sqr(F + yb));
                distance2 := sqrt(xbsquared + sqr(F - yb));
                B0 := 0.5 * (distance1 + distance2); {Parameter of the ellipse through (xb, yb)}
                C0 := 0.5 * (distance1 - distance2); {Parameter of the hyperbola through (xb, yb)}
                AA0 := sqrt(sqr(B0) - Fsquared); {x-axis intercept of the ellipse}
                D0 := sqrt(abs(Fsquared - sqr(C0))); {Remove abs() unless "172, 72" case.}

                xpp := B0scale * (B0mid - B0) + ImSize; {-63.0..63.0}

 {Index into fProj, to begin interpolation—round towards minus infinity--it = ImSizeOv2 for center of the image.}
                it := round(xpp - 0.5); {index for linear interpolation.}
                rit := it;

{Calculate the left-most point of the convolution, needed for linear interpolation.}
                t := it + 2;
                sum1 := 0.0;
                if (it >= 0) and (it <= npx - 1) then
                    for mm := 1 to it + 1 do
                        sum1 := sum1 + rh[t - mm] * wProj[mm - RaysPerProjOv2Plus1]
                else if (it >= npx) and (it <= 2 * npx - 1) then
                    for mm := (it - npx + 2) to npx do
                        sum1 := sum1 + rh[t - mm] * wProj[mm - RaysPerProjOv2Plus1]
                else
                    begin
                    writeln('Indexing error in the convolution');
                    readln;
                    end;

{Calculate the next point to the right in the convolution, needed for linear interpolation.}
                itplus1 := it + 1;
                t := itplus1 + 2;
                sum2 := 0.0;
                if (itplus1 >= 0) and (itplus1 <= npx - 1) then
                    for mm := 1 to itplus1 + 1 do
                        sum2 := sum2 + rh[t - mm] * wProj[mm - RaysPerProjOv2Plus1]
                else if (itplus1 >= npx) and (itplus1 <= 2 * npx - 1) then
                    for mm := (itplus1 - npx + 2) to npx do
                        sum2 := sum2 + rh[t - mm] * wProj[mm - RaysPerProjOv2Plus1]
                else
                    begin
                    writeln('Indexing error in the convolution');
                    readln;
                    end;
                weight := sqr(AA0 * F) / (sqr(AA0 * C0) + sqr(B0 * D0));
```

```
                image[i]^[j] := image[i]^[j] + weight * (sum1 + (xpp - rit) * (sum2 - sum1));
                end; {if}
            j := j + PixelSkip
            end;
        if Button = true then {Get a chance to quit.}
            begin
            write('Do you want to quit? [yes/no] ');
            readln(quit);
            if quit = yes then
                begin
                write('Do you want to update the output file? [yes/no] ');
                readln(UpdateFile);
                if UpdateFile = yes then
                    goto 99
                else
                    ExitToShell;
                end;
            end;
        i := i + PixelSkip;
        end;
    end;

99:
{Fix the outlying pixels which were set to -INF.}
for i := -ImSizeOv2 to ImSizeOv2 do
    for j := -ImSizeOv2 to ImSizeOv2 do
        if image[i]^[j] = -INF then
            image[i]^[j] := 0.0;

{Prepare to write image to output file.}
close(inFile);

{Find how many non-zero pixels, in case of low-res.}
PixelCount := 0;
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    PixelCount := PixelCount + 1;
    i := i + PixelSkip
    end;
write(outFile, PixelCount); {Actual image size, in pixels.}
write(outFile, PixelCount); {Square image}

{Write the image to the output file.}
i := -ImSizeOv2;
while i <= ImSizeOv2 do
    begin
    j := -ImSizeOv2;
    while j <= ImSizeOv2 do
        begin
        write(outFile, image[i]^[j]);
        j := j + PixelSkip
        end;
    i := i + PixelSkip
    end;
close(outFile);

{Print out exit info.}
if WantWindow = yes then
    writeln('Hamming window');
```

```
GetDateTime(FinTime);
writeln('Execution time was: ', (FinTime - BegTime) / 60.0 : 5 : 1, ' minutes.');
rewrite(TimeToRunFile, 'HD:TimeToRunFile');
writeln(TimeToRunFile, 'Execution time was: ', (FinTime - BegTime) / 60.0 : 5 : 1, ' minutes.');
close(TimeToRunFile);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
SysBeep(1);
writeln('Input file: ', inFileName);
writeln('Output file: ', outFileName);
writeln('Trajectory file: ', TrajectoryFileName);
writeln('Overall weight: ', OverallWeight : 13 : 3);

if (WantShutDown = yes) and (not Button) then
    ShutDwnPower {Make sure all updated files are closed first.}
else
    pause('Done');
end.
```

REFERENCES

[1] John J. Kovaly, *Synthetic Aperture Radar.* Dedham, MA: Artech House, Inc., 1976.

[2] Charles Elachi, *Spaceborne Radar Remote Sensing* . New York, NY: The Institute of Electrical and Electronics Engineers, Inc., 1988.

[3] Charles Elachi, T. Bicknell, Rolando L. Jordan, and Chialin Wu, "Spaceborne synthetic aperture imaging radars: applications, techniques, and technology," *Proceedings of the IEEE*, vol. 70, pp. 1174–1209, 1982.

[4] H. Jensen, L. C. Graham, L. J. Porcello, and E. N. Leith, "Side-looking airborne radar," *Scientific American*, vol. 237, pp. 84-95, October 1977.

[5] Dale A. Ausherman, Adam Kozma, Jack L. Walker, Harrison M. Jones, and Enrico C. Poggio, "Developments in imaging radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-20, pp. 363-398, July 1984.

[6] C. Boesswetter, "MM wave SAR sensor design: Concept for an airborne low level reconnaissance system," *Multifunction Radar for Airborne Applications* , Advisory Group for Aerospace Research & Development, AGARD Conference Proceedings No. 381, (a publication of the North Atlantic Treaty Organization), Seine, France, pp. 2–4, 1986.

[7] R. Colin Johnson and Tom J. Schwartz, "DARPA backs neural nets," *Electronic Engineering Times*, p. 1, August 8, 1988.

[8] The NBC Nightly News, April 7, 1989.

[9] Tim Carrington and John Walcott, "Pentagon lags in race to match the Soviets in rocket launchers," *The Wall Street Journal*, p. 1, July 12, 1988.

[10] G. N. Tsandoulas, "Space-based radar," *Science*, vol. 237, pp. 257-262, July 17, 1987.

[11] "Satellite with 150-ft. span set for launch on Mission 27," *Aviation Week & Space Technology*, p. 25, November 7, 1988.

[12] Bob Davis, "Shuttle Atlantis prepares to return after putting spy satellite into orbit," *The Wall Street Journal*, back page, December 3, 1988.

[13] NBC Nightly News, December 2, 1988.

[14] Jim Van Nostrand, "EOS platform to train four eyes on Earth," *Electronic Engineering Times*, p. 92, February 20, 1989.

[15] "NASA Facts: Magellan," NF-143. A publication of the National Aeronautics and Space Administration, September 1985.

[16] A. Papoulis, *Signal Analysis.* New York, NY: McGraw-Hill, 1977.

[17] D. E. Vakman, *Sophisticated Signals and the Uncertainty Principle in Radar* , trans. K. N. Trirogoff. New York, NY: Springer-Verlag, 1968.

[18] August W. Rihaczek, *Principles of High-Resolution Radar*. New York, NY: McGraw-Hill, 1969.

[19] Chung-Ching Chen and Harry C. Andrews, "Target-motion induced radar imaging," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, pp. 2-14, January 1980.

[20] Donald R. Wehner, *High-Resolution Radar*. Norwood, MA: Artech House, 1987.

REFERENCES

[21] La Rue A. Hoffman, Keith H. Hurlbut, Dale E. Kind, and Herbert J. Wintroub, "A 94-GHz radar for space object identification," *IEEE Transactions on Microwave Theory and Techniques*, vol. MTT-17, pp. 1145-1149, December 1969.

[22] Stanley R. Deans, *The Radon Transform and Some of Its Applications* . New York, NY: Wiley, 1983.

[23] Chung-Ching Chen and Harry C. Andrews, "Multifrequency imaging of radar turntable data," *IEEE Transactions on Aerospace and Electronic Systems* , vol. AES-16, pp. 15-22, January 1980.

[24] N. H. Farhat, C. L. Werner, and T. H. Chu, "Prospects for three-dimensional projective and tomographic imaging radar networks," *Radio Science* , vol. 19, pp. 1347-1355, September-October 1984.

[25] John C. Kirk, "A discussion of digital processing in synthetic aperture radar," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, pp. 326-337, May 1975.

[26] David C. Munson, Jr., James Dennis O'Brien, and W. Kenneth Jenkins, "A tomographic formulation of spotlight-mode synthetic aperture radar," *Proceedings of the IEEE* , vol. 71, pp. 917-925, August 1983.

[27] J. L. Bauck and W. K. Jenkins, "Tomographic processing of spotlight-mode synthetic aperture radar signals with compensation for wavefront curvature," *Proceedings of the 1988 IEEE International Conference on Acoustics, Speech, and Signal Processing* , New York, NY, April 1988.

[28] J. L. Bauck and W. K. Jenkins, "Convolution-backprojection image reconstruction for bistatic synthetic aperture radar with correction for wavefront curvature and propagation attenuation," *Proceedings of the SPIE Technical Symposia on Aerospace Sensing* , Orlando, FL, March 1989.

[29] J. L. Bauck and W. K. Jenkins, "Convolution-backprojection image reconstruction for bistatic synthetic aperture radar," *Proceedings of the IEEE 1989 International Symposium on Circuits and Systems*, Portland, OR, May 1989.

[30] M. Bertero, C. De Mol, and E. R. Pike, "Linear inverse problems with discrete data. I: general formulation and singular system analysis," *Inverse Problems*, vol. 1, pp. 301-330, 1985.

[31] John W. Dettman, *Mathematical Methods in Physics and Engineering* , 2nd. ed. New York, NY: McGraw-Hill, 1969.

[32] Mita Dinkarrai Desai, "A new method of synthetic aperture radar image reconstruction using modified convolution back-projection algorithm," Ph. D. dissertation, University of Illinois, Urbana, IL, 1985.

[33] Johann Radon, "Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten [On the determination of functions from their integrals along certain manifolds]," *Berichte Sächsische Akademie der Wissenschaften. Leipzig, Math.—Phys. Kl.*, vol. 69, pp. 262-267, 1917 (in German).

[34] Gabor T. Herman, *Image Reconstruction From Projections: The Fundamentals of Computerized Tomography*. New York, NY: Academic Press, 1980.

REFERENCES

[35] Avinash C. Kak and Malcolm Slaney, *Principles of Computerized Tomographic Imaging* . New York, NY: IEEE Press, 1988.

[36] Avinash C. Kak, "Tomographic imaging with diffracting and nondiffracting sources," in *Array Signal Processing* , Simon Haykin, Ed. Englewood Cliffs, NJ: Prentice-Hall, 1985, pp. 351-428.

[37] Dan E. Dudgeon and Russell M. Mersereau, *Multidimensional Digital Signal Processing* . Englewood Cliffs, NJ: Prentice-Hall, 1984.

[38] Russell M. Mersereau and Alan V. Oppenheim, "Digital reconstruction of multidimensional signals from their projections, *Proceedings of the IEEE* , vol. 62, pp. 1319-1338, October 1974.

[39] Keigo Iizuka, *Engineering Optics*, 2nd ed. Berlin: Springer-Verlag, 1987.

[40] *Proceedings of the IEEE* , vol. 71, March 1983 (Special Issue on Computerized Tomography).

[41] Henry J. Scudder, "Introduction to computer aided tomography," *Proceedings of the IEEE* , vol. 66, pp. 628-637, June 1978.

[42] Henry Stark, John W. Woods, Indraneel Paul, and Rajesh Hingorani, "Direct Fourier reconstruction in computer tomography," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, pp. 237-245, April 1981.

[43] Hong Fan and Jorge L. C. Sanz, "Comments on 'Direct Fourier reconstruction in computer tomography,'" *IEEE Transactions on Acoustics, Speech, and Signal Processing* , vol. ASSP-33, pp. 446-449, April 1985.

[44] Bruce Clatworthy Mather, "Fourier domain interpolation methods for synthetic aperture radar", Ph. D. dissertation, University of Illinois, Urbana, IL, 1986.

[45] David Alan Hayner, "The missing cone problem in computer tomography and a model for interpolation in synthetic aperture radar," Ph. D. dissertation, University of Illinois, Urbana, IL, 1983.

[46] Paul A. Rattey and Allen G. Lindgren, "Sampling the 2-D Radon transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing* , vol. ASSP-29, pp. 994-1002, August 1981.

[47] C. H. Chapman and P. W. Cary, "The circular harmonic Radon transform," *Inverse Problems*, vol. 2, pp. 23-49, 1986.

[48] Eric W. Hansen, "Theory of circular harmonic image reconstruction," *Journal of the Optical Society of America*, vol. 71, pp. 304-308, March 1981.

[49] Eric W. Hansen, "Circular harmonic image reconstruction: experiments," *Applied Optics* , vol. 20, pp. 2266-2274, July 1981.

[50] Barry Paul Medoff, "Image reconstruction from limited data," Ph. D. dissertation, Stanford University, Stanford, CA, 1983.

[51] Barry P. Medoff, "An inner product framework for image reconstruction," *Proceedings of the 1985 IEEE International Conference on Acoustics, Speech, and Signal Processing* , paper no. 28.6.1, pp. 1073-1076.

REFERENCES

[52] Barry P. Medoff, "Image reconstruction from limited data: theory and applications in computerized tomography," in *Image Recovery: Theory and Application*. New York, NY: Academic Press, 1987, pp. 321-368.

[53] Jack Sklansky, "On the Hough technique for curve detection," *IEEE Transactions on Computers*, vol. C-27, pp. 923-926, October 1978.

[54] Stephen D. Shapiro, "Feature space transforms for curve detection," *Pattern Recognition*, vol. 10, pp. 129-143, 1978.

[55] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, pp. 111-122, 1981.

[56] Robert L. Visentin, "A digital signal processing view of strip-mapping synthetic aperture radar," M.S. thesis, University of Illinois, Urbana, IL,1988.

[57] David C. Munson, Jr. and Robert L. Visentin, "A signal processing view of strip-mapping synthetic aperture radar," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-37, December 1989 (to appear).

[58] Gordon J. A. Bird, *Radar Precision and Resolution*. New York, NY: Wiley, 1974.

[59] Robert O. Harger, *Synthetic Aperture Radar Systems*. New York, NY: Academic Press, 1970.

[60] J. Patrick Fitch, *Synthetic Aperture Radar*. New York, NY: Springer-Verlag, 1988.

[61] Eli Brookner, Ed., *Radar Technology*. Dedham, MA: Artech House, 1977.

[62] Nadav Levanon, *Radar Principles*. New York, NY: Wiley, 1988.

[63] William M. Brown and Leonard J. Porcello, "An introduction to synthetic-aperture radar," *IEEE Spectrum*, vol. 6, pp. 52-62, September 1969.

[64] A. B. E. Ellis, "The processing of synthetic aperture radar signals," *GEC Journal of Research*, vol. 2, pp. 169-177, no. 3, 1984. Also published in *The Radio and Electronic Engineer*, vol. 53, pp. 107-114, March 1983.

[65] Jack L. Walker, "Range-Doppler imaging of rotating objects," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, pp. 23-52, January 1980.

[66] William M. Brown, "Walker model for radar sensing of rigid target fields," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, pp. 104-107, January 1980.

[67] Merrill I. Skolnik, *Introduction to Radar Systems*. New York, NY: McGraw-Hill, 1980.

[68] J. R. Klauder, A. C. Price, S. Darlington, and W. J. Albersheim, "The theory and design of chirp radars," *The Bell System Technical Journal*, vol. 39, pp. 745-808, July 1960.

[69] Athanasios Papoulis, *Probability, Random Variables, and Stochastic Processes*. New York, NY: McGraw-Hill, 1984.

[70] Alan Di Cenzo, "Digital synthetic aperture radar processing via optical computers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-25, pp. 429-433, May 1989.

[71] G. Tricoles and Nabil H. Farhat, "Microwave holography: applications and techniques," *Proceedings of the IEEE*, vol. 65, pp. 108-121, January 1977.

REFERENCES

[72] Chi K. Chan and Nabil H. Farhat, "Frequency swept tomographic imaging of three-dimensional perfectly conducting objects," *IEEE Transactions on Antennas and Propagation* , vol. AP-29, pp. 312-319, March 1981.

[73] N. H. Farhat, C. Yi Ho, and Li Szu Chang, "Projection theorems and their application in multidimensional signal processing," *Proeedings SPIE*, vol. 388, pp. 140-149, 1983.

[74] Bernard D. Steinberg, "Microwave imaging of aircraft," *Proceedings of the IEEE* , vol. 76, pp. 1578-1592, December 1988.

[75] Bernard D. Steinberg, *Microwave Imaging with Large Antenna Arrays: Radio Camera Principles and Techniques*. New York, NY: Wiley, 1983.

[76] Shalom Halevy, "Tomographic radar imaging techniques," *Proceedings of the 22nd Asilomar Conference on Circuits, Systems, and Computers* , Pacific Grove, CA, October-November 1988.

[77] M. Bernfeld, "CHIRP Doppler radar," *Proceedings of the IEEE*, vol. 72, pp. 540-541, April 1984.

[78] Ephraim Feig and F. Alberto Grünbaum, "Tomographic methods in range-Doppler radar," *Inverse Problems*, vol. 2, pp. 185-195, 1986.

[79] Donald L. Snyder, Harper J. Whitehouse, J. Trent Wohlschlaeger, and Robert C. Lewis, "A new approach to radar/sonar imaging," 1986 SPIE Conference on Advanced Algorithms and Architectures, *Proceedings SPIE*, vol. 696, no. 20, San Diego, CA, 1986.

[80] *IEEE Transactions on Sonics and Ultrasonics*, Special Issue on Digital Acoustical Imaging, vol. SU-31, July 1984.

[81] Hua Lee and Glen Wade, Eds., *Imaging Technology*. New York, NY: IEEE Press, 1986.

[82] David C. Munson, Jr., and Jorge L. C. Sanz, "Image reconstruction from frequency-offset Fourier data," *Proceedings of the IEEE*, vol. 72, pp. 661-669, June 1984.

[83] Dag T. Gjessing, *Target Adaptive Matched Illumination Radar: Principles and Applications*. London: Peter Peregrinus Ltd., 1986.

[84] Berthold K. P. Horn, "Density reconstruction using arbitrary ray-sampling schemes," *Proceedings of the IEEE*, vol. 66, pp. 551-562, May 1978.

[85] E. V. Jull, *Aperture Antennas and Diffraction Theory* . Stevanage, UK: Peter Peregrinus Ltd. on behalf of the Institution of Electrical Engineers, 1981.

[86] D. G. Luenberger, *Optimization by Vector Space Methods*. New York, NY: Wiley, 1969.

[87] W. Thomas Cathey, *Optical Information Processing and Holography* . New York, NY: Wiley, 1974.

[88] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and Willian T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*. Cambridge, MA: Cambridge University Press, 1986.

[89] Richard G. Wiley, *Electronic Intelligence: The Interception of Radar Signals* . Dedham, MA: Artech House, 1985.

[90] A. Bruce Carlson, *Communication Systems: An Introduction to Signals and Noise in Electrical Communication*. New York, NY: McGraw-Hill, 1975.

REFERENCES

[91] D. O. North, "An analysis of the factors which determine signal-noise discrimination in pulsed carrier systems," RCA Lab. Rept. PTR-6C, June 1943.

[92] W. J. Caputi, "Stretch: a time-transformation technique," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-7, no. 2, pp. 269-278, March 1971.

[93] C. E. Cook and M. Bernfeld, *Radar Signals*. New York, NY: Academic Press, 1967.

[94] Warren L. Stutzman and Gary A. Thiele, *Antenna Theory and Design* . New York, NY: Wiley, 1981.

VITA

Jerald Lee Bauck was born in Western Kansas in 1955. He holds a B.S. from Kansas State University (Honors Program), Magna Cum Laude, and an M.S. from the University of Illinois, both in Electrical Engineering. Mr. Bauck worked for Motorola's Government Electronics Group in Scottsdale, Arizona, for five years where he earned the Motorola Engineering Award. He then returned to the University of Illinois to study towards the doctorate. While there, he was designated an Outstanding Teacher as a graduate teaching assistant and was also a research assistant. He is a member of the Institute of Electrical and Electronics Engineers, the Audio Engineering Society, Eta Kappa Nu, Tau Beta Pi, and Phi Kappa Phi. His ten journal and conference papers are in the areas of radar, antennas, acoustics, and audio, while 13 technical reports also cover pattern classification, acoustic direction finding, Doppler tracking, and signal processing. He holds six U.S. patents.