UNIVERSIDADE ESTADUAL DE CAMPINAS

Faculdade de Engenharia Elétrica e de Computação

JULIANA DE ALMEIDA GÓES

TECHNIQUES FOR HIGH-RESOLUTION 3D IMAGES WITH SYNTHETIC APERTURE RADAR

TÉCNICAS PARA OBTER IMAGENS 3D DE RESOLUÇÃO FINA COM RADAR DE ABERTURA SINTÉTICA

CAMPINAS

2022

JULIANA DE ALMEIDA GÓES

TECHNIQUES FOR HIGH-RESOLUTION 3D IMAGES WITH SYNTHETIC APERTURE RADAR

TÉCNICAS PARA OBTER IMAGENS 3D DE RESOLUÇÃO FINA COM RADAR DE ABERTURA SINTÉTICA

Thesis presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering, in the area of Telecommunications and Telematics.

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutora em Engenharia Elétrica, na área de Telecomunicações e Telemática.

Supervisor/Orientador: HUGO ENRIQUE HERNANDEZ FIGUEROA

Co-supervisor/Coorientador: LEONARDO SANT'ANNA BINS

Este trabalho corresponde à versão final da tese defendida pela aluna Juliana de Almeida Góes, orientada pelo Prof. Dr. Hugo Enrique Hernandez Figueroa e coorientada pelo Dr. Leonardo Sant'Anna Bins.

CAMPINAS

2022

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

G553t Góes, Juliana de Almeida de, 1987-
Techniques for high-resolution 3D images with synthetic aperture radar / Juliana de Almeida Góes. – Campinas, SP : [s.n.], 2022.

Orientador: Hugo Enrique Hernandez Figueroa.
Coorientador: Leonardo Sant'Anna Bins.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Radar de abertura sintética. 2. Processamento de sinais. 3. Imagem tridimensional. 4. Trajetórias de voo. 5. Aeronaves não tripuladas. I. Hernandez-Figueroa, Hugo Enrique, 1959-. II. Bins, Leonardo Sant'Anna, 1961-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Técnicas para obter imagens 3D de resolução fina com radar de abertura sintética
Palavras-chave em inglês:
Synthetic aperture radar
Signal processing
3D imaging
Flight paths
Drone aircraft
Área de concentração: Telecomunicações e Telemática
Titulação: Doutora em Engenharia Elétrica
Banca examinadora:
Hugo Enrique Hernandez Figueroa [Orientador]
João Roberto Moreira Neto
Renato Machado
Yusef Rafael Cáceres Zuñiga
Gustavo Fraidenraich
Data de defesa: 17-08-2022
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0003-0832-9823
- Currículo Lattes do autor: http://lattes.cnpq.br/0094252806969282

COMISSÃO JULGADORA – TESE DE DOUTORADO

Candidata: Juliana de Almeida Góes **RA:** 190738

Data da defesa: 17 de agosto de 2022

Título da Tese: "Techniques for High-Resolution 3D images with Synthetic Aperture Radar"

Prof. Dr. Hugo Enrique Hernandez Figueroa (Presidente)

Prof. Dr. João Roberto Moreira Neto

Prof. Dr. Renato Machado

Prof. Dr. Yusef Rafael Cáceres Zuñiga

Prof. Dr. Gustavo Fraidenraich

A Ata de Defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de Pós-Graduação da Faculdade de Engenharia Elétrica e de Computação.

ACKNOWLEDGMENTS

First and foremost, I thank God for the endless blessings and precious opportunities He has given me. I am grateful for my parents' love and support, without which I would never have come this far. Also, I am thankful to my brother Daniel for his friendship and helpful insights.

I want to express my gratitude to my advisor Hugo Enrique Hernandez Figueroa, for his guidance throughout my doctoral studies. Likewise, I am extremely grateful to my co-advisor, Leonardo Sant'Anna Bins, for his patience and enthusiasm in walking me through my research project. Also, I would like to extend my sincere thanks to João Roberto Moreira Neto for his valuable assistance.

I want to thank all my colleagues from the SAR research group, with special thanks to Valquíria, Felício and Luciano. It was a pleasure working with you all. Thanks should also go to Radaz Ltda. A major part of this thesis would not have been possible without the drone SAR data provided by them.

Finally, I would like to acknowledge that this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

ABSTRACT

Drone-based Synthetic Aperture Radar (SAR) is a technology on the rise, though SAR is a well-established technology for aircraft and satellite systems. This thesis focuses on developing techniques to acquire high-resolution 3D SAR images from a drone-borne SAR system. First, this thesis developed a Fast Factorized Back Projection (FFBP) algorithm with two original features, which is suitable for processing 3D images from data acquired by non-linear flight paths. While maintaining good focusing quality, the FFBP algorithm was up to 21 times faster than the Back Projection (BP) algorithm when processing 3D images and up to 13 times faster for 2D images. Second, this thesis introduced a method for designing spiral flight paths for 3D SAR based on a new acquisition geometry proposed here. The proposed conical spiral flight path achieved a 1.76 m vertical resolution for a quad-corner reflector, while the state-of-the-art cylindrical spiral flight path achieved 2.33 m. Both contributions were tested with simulation results and validated with actual SAR data acquired by a drone-borne SAR system operating on the P-band.

Keywords: synthetic aperture radar; radar signal processing; 3D imaging; flight paths; drone aircraft.

RESUMO

O Radar de Abertura Sintética (SAR) é uma tecnologia bem estabelecida para sistemas de aeronaves e satélites. Por outro lado, o SAR embarcado em drone é uma tecnologia em ascensão. O foco desta tese é desenvolver técnicas para obter imagens SAR 3D de resolução fina com dados obtidos por um sistema SAR embarcado em drone. Primeiramente, desenvolveu-se um algoritmo *Fast Factorized Back Projection* (FFBP) com duas características originais, sendo adequado para processar imagens 3D a partir de dados adquiridos por trajetórias de voo não lineares. Mantendo a imagem bem focada, o algoritmo FFBP foi até 21 vezes mais rápido que o algoritmo *Back Projection* (BP) ao processar imagens 3D e até 13 vezes mais rápido para imagens 2D. Em segundo lugar, criou-se um método para projetar trajetórias espirais para SAR 3D baseado em uma nova geometria de aquisição proposta neste trabalho. A nova trajetória espiral cônica alcançou uma resolução vertical de 1,76 m para um refletor de quatro cantos, enquanto a trajetória espiral cilíndrica, que representa o estado-da-arte, atingiu 2,33 m. Ambas as contribuições foram testadas com resultados de simulação e validadas com dados SAR reais adquiridos por um sistema SAR embarcado em drone operando na banda P.

Palavras-chave: radar de abertura sintética; processamento de sinal de radar; imagem 3D; trajetórias de voo; aeronaves não tripuladas.

LIST OF FIGURES

List of Tables

List of Acronyms

BP	Back Projection
CI	Confidence Interval
CT	Computed Tomography
DEM	Digital Elevation Model
DTM	Digital Terrain Model
ECEF	Earth-Centered Earth-Fixed
ENU	East-North-Up
FFBP	Fast Factorized Back Projection
FFT	Fast Fourier Transform
GNSS	Global Navigation Satellite System
IFFT	Inverse Fast Fourier Transform
IMU	Inertial Measurement Unit
LOS	Line-of-Sight
MSS	Motion Sensing System
NED	North-East-Down
PSLR	Peak-to-Sidelobe Ratio
RMSE	Root-Mean-Square Error
SAR	Synthetic Aperture Radar
SE	Standard Error
SNR	Signal-to-Noise Ratio
SRTM	Shuttle Radar Topography Mission

LIST OF SYMBOLS

A_e	Effective area of the antenna
A_r	Amplitude of the chirped pulse
A_{x_n}	Dimension of the child subimage in the x direction
A_{y_n}	Dimension of the child subimage in the y direction
A_{z_n}	Dimension of the child subimage in the z direction
b	Baseline
B	Tomographic aperture length
$b_\perp$	Effective baseline
$B_\perp$	Effective tomographic aperture
$b_{\perp,c}$	Critical baseline
bw	Fractional bandwidth
c	Child subimage index or speed of light
$\overline{CP}_n$	Distance between a child subaperture and a parent subaperture
$\overline{CS}_{n,c}$	Distance between a child subaperture and a range sample in a child subimage
d	Index of the child subimages generated by one parent subimage
d_a	Antenna aperture length
D_n	Number of children generated by each parent at a given recursion
d_x	Index of the child subimages generated in the x direction
D_x	Number of divisions in the x direction
d_y	Index of the child subimages generated in the y direction
D_y	Number of divisions in the y direction
d_z	Index of the child subimages generated in the z direction
D_z	Number of divisions in the z direction
f	Frequency

f_D Doppler frequency or azimuth frequency

f_r Frequency of the chirped pulse

g Range-compressed data

G Antenna gain

H_{DEM} Terrain height

$H_{LOS_\perp}$ Maximum width perpendicular to the line-of-sight direction

$\boldsymbol{h}_m$ Pixel position

$\boldsymbol{h}_{n-1}$ Parent subimage center

$\boldsymbol{h}_n$ Child subimage center without topographic information

$\widetilde{\boldsymbol{h}}_{n,c}$ Child subimage center with topographic information

k Radar position index, child subaperture index or wavenumber

k_g Ground wavenumber

k_z Vertical wavenumber

$\vec{k}_g$ Ground component of the wave vector

$\vec{k}_z$ Vertical component of the wave vector

K_0 Number of radar root positions

$\mathbb{K}_m$ Synthetic aperture of a pixel

K_n Number of child subapertures at each recursion

l Index of the parent subapertures that are combined into a child subaperture

L Number of subapertures combined at each recursion

L_{SA} Synthetic aperture length

m Pixel index

M_n Number of range samples at each recursion

n Recursion index

N Number of recursions

N_p Number of pixels in the along-track direction

N_t Number of turns

$\mathcal{O}$	Big O notation
p	Parent subimage index
P	Number of radar pulses and linear size of the image
P_r	Received power
P_t	Transmitted power
$\overline{PS}_{n,c}$	Distance between a parent subaperture and a range sample in a child subimage
Q_n^{sa}	Number of child subapertures at each recursion
Q_n^{si}	Number of child subimages at each recursion
q_x	Recurrent sequence for the x direction
q_y	Recurrent sequence for the y direction
q_z	Recurrent sequence for the z direction
R	Slant range
$\tilde{R}$	Estimated slant range
$\boldsymbol{r}_0$	Set of radar root positions
$\hat{\boldsymbol{r}}_0$	Set of midpoints between radar root positions
R_0	Mean or closest range from the radar to the target
R_{delay}	Slant range of the first range bin
$\boldsymbol{r}_k$	Radar position
$R_{m,k}$	Slant range from the radar to a pixel
R_{min}	Closest range from the radar to the image
$\boldsymbol{r}_{n-1}$	Parent subaperture center
$\boldsymbol{r}_n$	Child subaperture center
$\mathcal{R}^2$	Coefficient of determination
s	Processed data
s_{n-1}	Parent data
s_n	Child data
s_r	Waveform of the chirped pulse

S_r	Spectrum of the chirped pulse
T	Pulse width of a modulated pulse
t	Slow-time
t_{max}	Time for completing the spiral flight path
V	Platform speed
V_0	Tangential speed
V_z	Vertical speed
V_ρ	Radial speed
W	Bandwidth
$\widetilde{W}$	Equivalent bandwidth
W_g	Effective bandwidth in the ground plane
w_H	Hamming window function
$w_{m,k}$	Window function
W_z	Effective bandwidth in the vertical direction
x_t	Target position on the x-axis
z_0	Mean height of the spiral flight path
$z_{2\pi}$	Height of ambiguity
z_{base}	Height at the base of the spiral flight path
z_H	Terrain height
z_{max}	Maximum height
z_{min}	Minimum height
z_{top}	Height at the top of the spiral flight path
$\frac{\partial \varphi}{\partial z}$	Phase-to-height sensitivity
α	Azimuth angle (spherical or cylindrical coordinate system)
β	Tilt angle
γ	Grazing angle
δ_a	Azimuth resolution

Δ_c	Child subimage diagonal or child subimage width
δ_g	Ground resolution (linear flight path)
δ_{gr}	Ground range spacing
δ_k	Child subaperture length
δ_k^0	Aperture spacing
$\delta_{LOS_\perp}$	Resolution perpendicular to the line-of-sight direction
δ_p	Pixel spacing
δ_r	Range resolution
δ_{sr}	Slant range spacing
δ_{xy}	Ground resolution (circular flight path)
δ_z	Vertical resolution
ΔB	Sampling distance
$\Delta B_\perp$	Effective sampling distance
$\Delta B_{\perp,c}$	Critical sampling distance
Δf	Spectral shift
Δf_g	Spectral shift in the ground plane
Δf_z	Spectral shift in the vertical direction
$\Delta\vec{k}$	Wave vector shift
Δk_g	Ground wavenumber shift
Δk_z	Vertical wavenumber shift
Δr	Change in radius (spherical coordinate system)
ΔR	Slant range difference
$\Delta\tilde{R}$	Range error
$\Delta\alpha$	Change in azimuth angle (spherical coordinate system)
$\Delta\varepsilon$	Change in elevation angle (spherical coordinate system)
$\Delta\rho$	Radius variation (cylindrical coordinate system)
$\Delta\tau$	Pulse width of a square pulse or after pulse compression

$\Delta\varphi$	Phase difference
$\Delta\psi$	Look-angle difference
θ_{3dB}	Antenna elevation beamwidth
θ_{a}	Antenna beamwidth in azimuth
θ_{axis}	Depression angle of the antenna axis
θ_{far}	Far-range depression angle
$\theta_{n,c}$	Angle at the vertex associated with a child subaperture
θ_{near}	Near-range look-angle
Θ_{SA}	Synthetic aperture angle
κ	Term proportional to the hypothetical phase error
κ_{1}	Hypothetical phase error term calculated at the first recursion
κ_{avg}	Hypothetical phase error term calculated with average values
κ_{max}	Hypothetical phase error term calculated with maximum values
λ	Wavelength
$\Lambda_{n,k}$	Set of parent subapertures associated with a child subaperture
$\nu_{m,k}$	Fractional range index
$\nu_{n,c}$	Fractional range index
ρ	Radius (cylindrical coordinate system)
ρ_{0}	Mean radius of the spiral flight path
ρ_{base}	Radius at the base of the spiral flight path
ρ_{ci}	Radius of constant illumination
ρ_{max}	Maximum radius
ρ_{min}	Minimum radius
ρ_{top}	Radius at the top of the spiral flight path
σ	Radar cross-section of the target
$\sigma_{\Delta\varphi}$	Standard deviation of the phase error
σ_{xyz}	Covariance matrix

τ	Fast-time or return time
φ	Phase of the return signal
φ_a	Phase of the backscattered echoes
$\varphi_{m,k}$	Phase compensation term
$\varphi_{n,c}$	Phase compensation term at each recursion
φ_r	Phase of the chirped pulse
$\tilde{\varphi}_c$	Phase compensation term at the last recursion
ψ	Look-angle
Ψ_{s_r}	Autocorrelation function of the chirped pulse
ψ_0	Mean look-angle
$\tilde{\psi}_0$	Equivalent look-angle
$\mathbf{\Omega}_0$	Union between the radar root positions and the midpoints between them

SUMMARY

1 INTRODUCTION

Synthetic Aperture Radar (SAR) is a coherent imaging radar. It is typically installed on a moving platform – such as satellites, aircraft, or drones – and provides high-resolution images. The origin of SAR dates back to 1951, when Carl Wiley of the Goodyear Aircraft Corporation first described the Doppler beam sharpening technique, now referred to as aperture synthesis [1]–[3]. The first successful SAR mission occurred in 1957. The radar was mounted on a C-46F aircraft, and the system also comprised a ground processing van. The total weight of the equipment was 700 lbs (317.5 kg). This mission was part of Project MICHIGAN, a research program at the University of Michigan sponsored by the US Army [4].

Spaceborne SAR was introduced three decades later with the launch of NASA's Seasat satellite in 1978. Seasat operated on the L-band, monitoring oceanographic phenomena. Its mission successfully demonstrated spaceborne SAR capabilities, though it lasted only 106 days due to a short circuit in the electrical system [5]. Since then, there have been about twenty SAR satellite missions from different space agencies worldwide [6]. SAR imagery has been used for myriad remote sensing applications, including Earth topography, earthquake and volcano monitoring, forest biomass, ice and glacier cover, and ocean currents, to name a few. Moreover, data can be acquired day or night, independent of cloud coverage [2].

Another three decades passed until the rise of drone-borne SAR. One of the first successful experiments was reported in 2009 by researchers from the Technical University of Catalonia, Barcelona, Spain. They used a fixed-wing drone with a 2.5 m wingspan that had 45 min autonomy with a 5 kg payload. This SAR system operated in the C-band (5.3 GHz), transmitting up to 2W. The first version of their SAR equipment weighed 2.5 kg and fitted inside a 15 × 25 × 9 cm^3 box [7]. SAR drones are low-cost systems suitable for surveying small areas, opening new market possibilities [8].

1.1 THE DRONE-BORNE SAR SYSTEM

This thesis uses data from the drone-borne SAR system depicted in Figure 1.1. The system operates in three bands – C, L and P – and has four antennas: two interferometric C-band antennas, one L-band antenna with two polarizations, and one P-band antenna. Each operating band has its particularities, which can be combined to produce further information. In [9], the drone-borne SAR system estimated the forest height by comparing the C- and P-band height profiles. While the forest is transparent to the P-band, the canopy reflects the C-band. In [10], data from the three bands were put together to calculate sugarcane biomass, which was then used to predict the harvest date and productivity. Finally, in [11], L-band data were used to generate growth maps for a coffee crop.

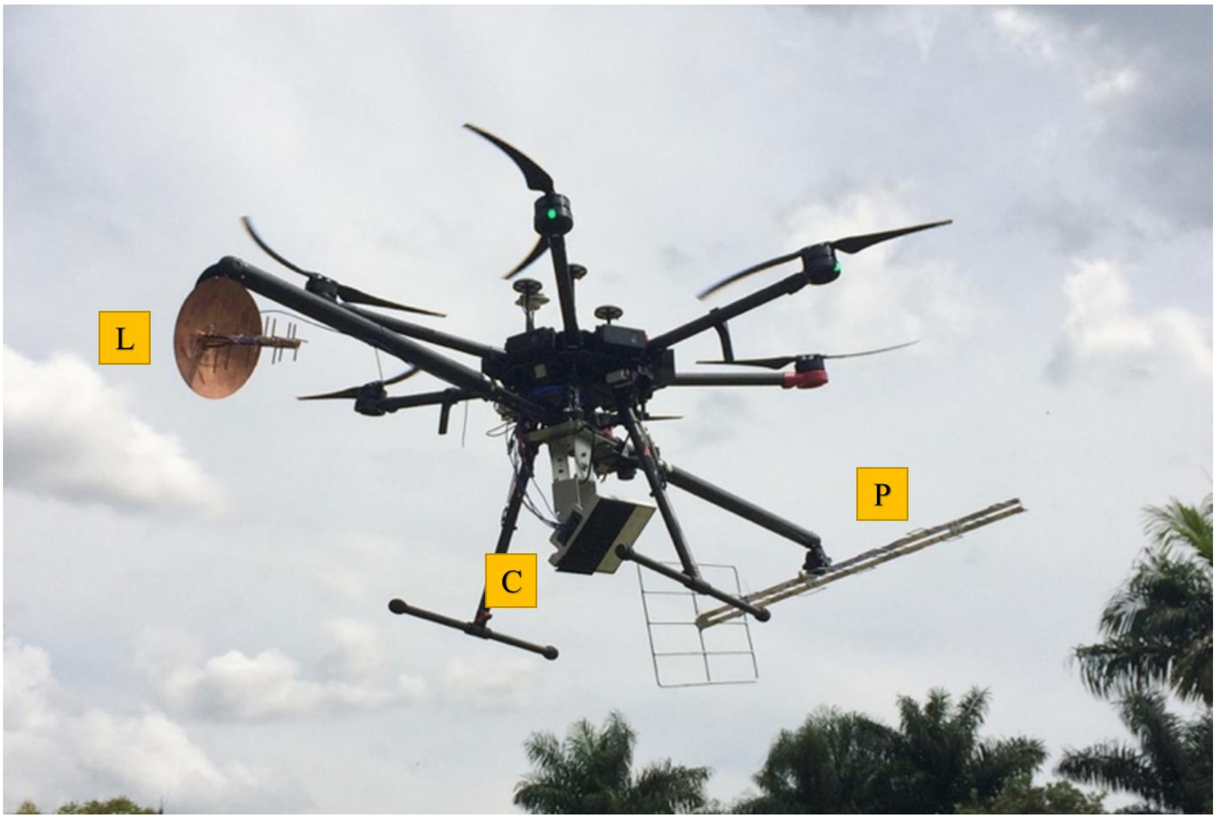


Figure 1.1 –The drone-borne Synthetic Aperture Radar (SAR) system equipped with the L-band, C-band, and P-band antennas. Source: Modified from Moreira *et al.* [8] © 2019 IEEE.

The multi-rotor drone in Figure 1.1 also carries a motion sensing system (MSS) comprising a global navigation satellite system (GNSS) and an inertial measurement unit (IMU). The MSS is accurate thanks to the IMU and because a ground station provides differential GNSS processing. In particular, the absolute position error is 15 cm, and the relative position error is less than 1 cm. Furthermore, in [12], corner reflectors were used to assess the system’s accuracy when producing deformation maps. The standard deviation of the resulting error was 4-7.4 mm.

In the research mentioned earlier, the drone-borne SAR system performed either linear or circular flight paths for producing 2D images and topographic maps. The data were processed using range-Doppler or Back Projection (BP) algorithms. Range-Doppler is an efficient algorithm best suited for linear flight paths, while BP has a high computing cost but works for any flight path.

1.2 THE THESIS

This thesis focuses on techniques to obtain high-resolution 3D SAR images. First, the multi-rotor drone has much more flexible flight mechanics than aircraft. Such flexibility allows us to explore new flight paths and thus achieve better resolutions than the state-of-the-art. Therefore, this thesis aims to create a method for designing flight paths from a vertical resolution requirement. Second, this investigation calls for a fast algorithm capable of processing 3D SAR images from arbitrary flight paths.

The purpose of these techniques is to enable subsurface tomography in future developments. There are several applications for subsurface tomography, such as humanitarian demining, archeology, and forensic investigation.

1.2.1 Contributions

This thesis presents a Fast Factorized Back Projection (FFBP) algorithm with two original features. First, it takes a data mapping approach that makes no assumptions about the flight path geometry, therefore being suitable for any flight path. Second, it employs a modified version of a classical tree structure, which is very flexible and can effortlessly manage 2D and 3D images.

Furthermore, the same SAR image can be processed with different setups, with a trade-off between phase error and processing time. So, this thesis tests the hypothesis that parameters at the start of processing can predict the standard deviation of the phase error at the output.

Moreover, this thesis introduces a new acquisition geometry that can perform better than the state-of-the-art: conical spiral flight paths. A method for designing such flight paths is also presented. This method depends on an analytical expression for the vertical resolution, which is also revised in this thesis.

1.2.2 Outline

Chapter 2 explains some fundamentals of SAR, starting from how radars work. Then, it describes what is recorded in the SAR data and how standard processing algorithms work. It also compares frequency-domain and time-domain algorithms.

Chapter 3 discusses FFBP, first reviewing the main strategies behind this type of algorithm. Then, the chapter details the proposed FFBP algorithm. Next, simulation results provide a proof of concept, showing that the FFBP algorithm works well in different conditions. Finally, a phase error analysis evaluating different setups is performed for actual SAR data acquired by the drone-borne SAR system.

Chapter 4 addresses spiral flight paths. First, it reviews the state-of-the-art acquisition geometries. Then, it explains the wavenumber shift concept, which is key to revising the vertical resolution expression. Next, the revised expression is validated through simulation results, while the influence of different parameters of the spiral flight path is also investigated. Finally, the method for designing spiral flight paths is proposed and validated with actual SAR data from the drone-borne SAR system.

Chapter 5 concludes this thesis. Lastly, the Appendix provides a parallel between the MATLAB code and the explanation of the FFBP algorithm in Chapter 3.

1.2.3 Publications

The subject of Chapter 3 yielded three publications: a conference paper [13], an article [14], and an open-access MATLAB code [15]. The findings detailed in Chapter 4 shall be published in the future.

2 SYNTHETIC APERTURE RADAR

Imaging radars prior to SAR typically used narrow beamwidth antennas to obtain suitable resolutions in azimuth (along-track). The beamwidth of an antenna (θ_a) can be estimated from the wavelength (λ) and the aperture length (d_a) [1]–[3]:

$$\theta_a \approx \frac{\lambda}{d_a} \tag{2.1}$$

The narrower the beamwidth, the finer the azimuth resolution would be. However, the azimuth resolution would also depend on the range from the radar to the target (R) [1]–[3]:

$$\delta_a = R\theta_a = R\frac{\lambda}{d_a} \tag{2.2}$$

So if a system operated with a 3 cm wavelength and the antenna aperture were 3 m long, the azimuth resolution at 10 km from the radar would be 100 m.

In contrast, SAR takes advantage of the platform movement and uses coherent processing to build a synthetic aperture far larger than the actual antenna aperture. Furthermore, in linear flight paths, the synthetic aperture increases with range. As a result, the azimuth resolution is no longer degraded as the range increases, and it is given by [1]–[3]:

$$\delta_a \approx \frac{d_a}{2} \tag{2.3}$$

Thus, for the same antenna aperture $d_a = 3$ m, the new azimuth resolution becomes 1.5 m, an impressive improvement over the previous example.

As the platform moves forward, the SAR system periodically transmits electromagnetic pulses, making each radar pulse correspond to a different azimuth position. After each pulse transmission, the radar takes several samples of the backscattered echo. These samples translate into different range distances traveled by the signal and contain both amplitude and phase information. The result is a complex raw data matrix with one dimension associated with the azimuth positions and another with the range samples (see Figure 2.1).

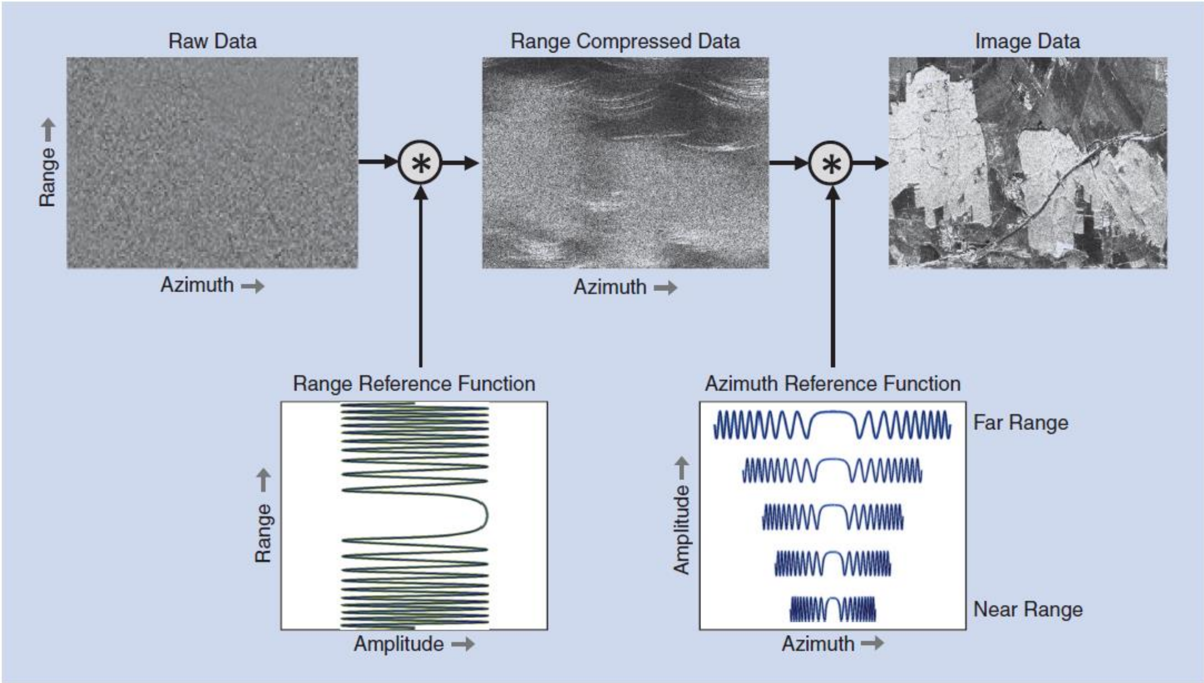


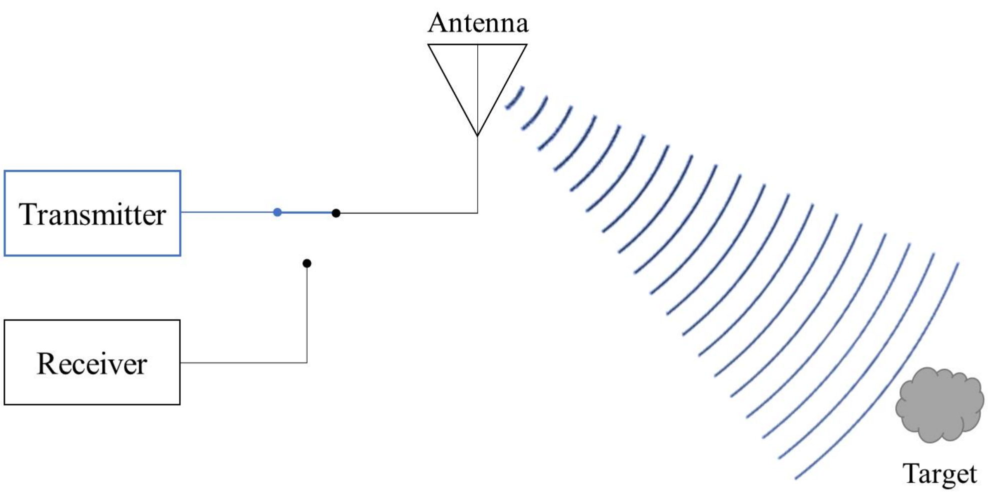
Figure 2.1 – Basic SAR processing steps: range compression and azimuth compression. Schematic for a frequency-domain algorithm. Source: Moreira et al. [2] © 2013 IEEE.

The SAR data must be processed before providing useful information. The first processing step, called range compression, consists of a matched filter operation using the transmitted pulse as a reference function. The second is called azimuth compression. Different algorithms can execute it: frequency-domain algorithms perform another matched filter operation in the azimuth direction (see Figure 2.1), while time-domain algorithms execute a coherent integration.

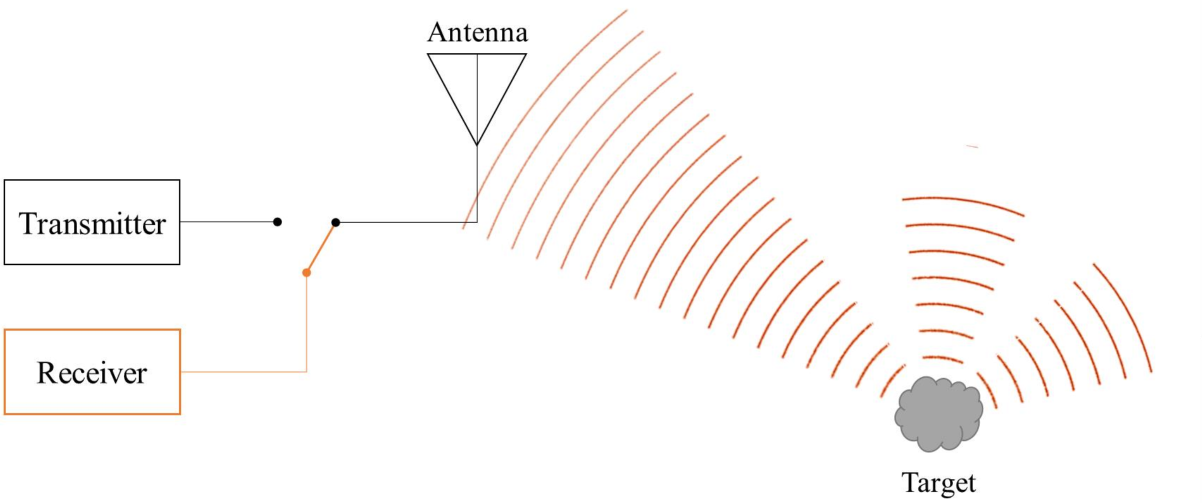
After a brief introduction to radar fundamentals in Section 2.1, the following sections cover the processes and elements illustrated in Figure 2.1. Section 2.2 addresses range compression and the range reference function. Section 2.3 describes the azimuth reference function, then presents an overview of frequency-domain algorithms and their limitations. Finally, Section 2.4 details a key time-domain algorithm.

2.1 WHAT IS RADAR?

Radar – radio detection and ranging – is an electromagnetic system for detecting and locating targets. It works by transmitting a signal (see Figure 2.2(a)), receiving its backscattered echo (see Figure 2.2(b)), and then extracting information from it. The most basic information is the time it takes for the echo to return. Since the signal speed is known, the return time estimates the range between the radar and the target.



(a)



(b)

Figure 2.2 – Radar operation: (a) signal transmission and (b) reception of the backscattered echo.

Figure 2.2 shows a monostatic radar, which uses the same antenna for transmission and reception. If the transmitting and receiving antennas were separate, it would be a bistatic radar. However, bistatic operations shall not be covered in this thesis.

2.1.1 Fundamental Relationships

In radar operations, the signal travels from the radar to the target and back again (two-way range distance). In addition, the signal propagates at the speed of light (c). Therefore, if τ is the return time of the backscattered echo and R is the range between the radar and the target, then [3]:

$$\tau = \frac{2R}{c} \tag{2.4}$$

Another fundamental relationship is between the range and the return phase (φ). Each time the signal moves one wavelength (λ), its phase completes a 2π cycle. Thus, φ is given by [3]:

$$\varphi = -\frac{4\pi R}{\lambda} \tag{2.5}$$

The negative sign denotes a phase delay.

2.1.2 Radar Equation

The radar equation is a valuable tool for designing radar systems. It calculates how much of the transmitted power is reflected by the target and received at the radar [1], [3]:

$$P_r = P_t G \times \frac{\sigma}{4\pi R^2} \times \frac{A_e}{4\pi R^2} \tag{2.6}$$

The three factors on the right-hand side of (2.6) represent the physical processes that take place (see Figure 2.2):

- The power that leaves the radar antenna is given by the transmitted power (P_t) and the antenna gain (G);
- When the signal arrives at the target, the power has been distributed on a sphere of radius R. The amount reflected back to the radar is determined by the target's radar cross-section (σ), which is measured in units of area;

- When the backscattered signal is received at the radar, it has been further attenuated by $4\pi R^2$. Also, the received power (P_r) depends on the antenna's effective area (A_e).

The effective area of an antenna is related to the antenna gain by [3]:

$$A_e = \frac{G\lambda^2}{4\pi} \tag{2.7}$$

Therefore, (2.6) can be rewritten as [3]:

$$P_r = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 R^4} \tag{2.8}$$

2.1.3 Radar Frequency Bands

Radar systems use a wide frequency spectrum from 30 MHz to 110 GHz [16], which can be conveniently grouped into bands. However, since these bands were defined during World War II, the chosen naming scheme was not immediately obvious for security reasons. Thus, the resulting letter designations have no apparent logic [3].

Nonetheless, band letter designations are still used today and have become accepted practice amongst radar engineers. Letter designations can describe the band of operation without stating the frequency limits in numerical terms, such as in titles of published papers or advertising radar systems and components. In addition, letter designations can communicate concisely a whole set of characteristics, applications, and environmental constraints that distinguish each particular band [16].

Table 2.1 shows the frequency bands used for SAR remote sensing, using letter designations common to the radar community. Note that the actual frequencies allocated for radar use by the International Telecommunications Union are narrower bands within this broad classification [5].

The drone-borne SAR system presented in Section 1.1 works in three bands: C, L, and P. Because the P-band has ground penetrating capabilities, it is suitable for SAR tomography. Therefore, only P-band results shall be presented in this thesis.

Table 2.1 – Letter designation of radar bands used for SAR remote sensing. Source: Modified from NASA SAR Handbook, 2019 [5].

Band	Frequency [GHz]	Wavelength [cm]	Typical application
P	0.3 – 1	100 – 30	Biomass; vegetation mapping and assessment. Experimental SAR.
L	1 – 2	30 – 15	Medium resolution SAR (Geophysical monitoring; biomass and vegetation mapping; high penetration; SAR interferometry)
S	2 – 4	15 – 7.5	Little but increasing use for SAR-based Earth observation; agriculture monitoring
C	4 – 8	7.5 – 3.8	SAR workhorse (global mapping; change detection; monitoring of areas with low to moderate vegetation; improved penetration; higher coherence); Ice, ocean, maritime navigation
X	8 – 12	3.8 – 2.4	High-resolution SAR (urban monitoring; ice and snow, little penetration into vegetation cover; fast coherence decay in vegetated areas)
Ku	12 – 18	2.4 – 1.7	Rarely used for SAR (satellite altimetry)
K	18 – 27	1.7 – 1.1	Rarely used (H_2O absorption)
Ka	27 – 40	1.1 – 0.8	Rarely used for SAR (airport surveillance)

2.2 FAST-TIME AXIS

The range axis of the SAR data matrices (see Figure 2.1) contains samples of the return signal, which travels at the speed of light. Typical sampling rates are a few tens of MHz or higher. Thus, the range axis is also called the fast-time axis [2].

2.2.1 Chirped Pulse

2.2.1.1 Range resolution

The radar range resolution is given by its ability to separate the backscattered echoes of two distinct point targets. For a square pulse, the limit condition is when the ending of the first echo coincides with the beginning of the second. In this case, the difference in returning time equals the pulse width ($\Delta\tau$). Therefore, the range resolution is [1], [3]:

$$\delta_r = \frac{c\Delta\tau}{2} \tag{2.9}$$

However, square pulses are not practical for SAR applications. For example, suppose a SAR system requires a range resolution of $\delta_r = 1.5$ m. Then, the pulse width would need to be $\Delta\tau$ = 10 ns, which is highly impractical (see Figure 2.3(a)). The radar would have to transmit a prohibitive peak power during $\Delta\tau$ and then remain silent for a much longer period until the next pulse transmission.

The solution is to employ pulse compression, illustrated in Figure 2.3(b,c). The radar transmits a modulated pulse of width T and bandwidth W, with $T \gg 1/W$. Nevertheless, after pulse compression, the width of the main lobe becomes $\Delta\tau \approx 1/W$. Moreover, the transmit energy is distributed over T, so the peak power is no longer an issue. Then, the range resolution becomes [1], [3]:

$$\delta_r = \frac{c}{2W} \tag{2.10}$$

For the previous example, a range resolution of δ_r =1.5 m would require a $W =$ 100 MHz bandwidth. In addition, suppose that $T = 20$ μs, then the resulting peak power would be 2000 times lower for the same transmit energy.

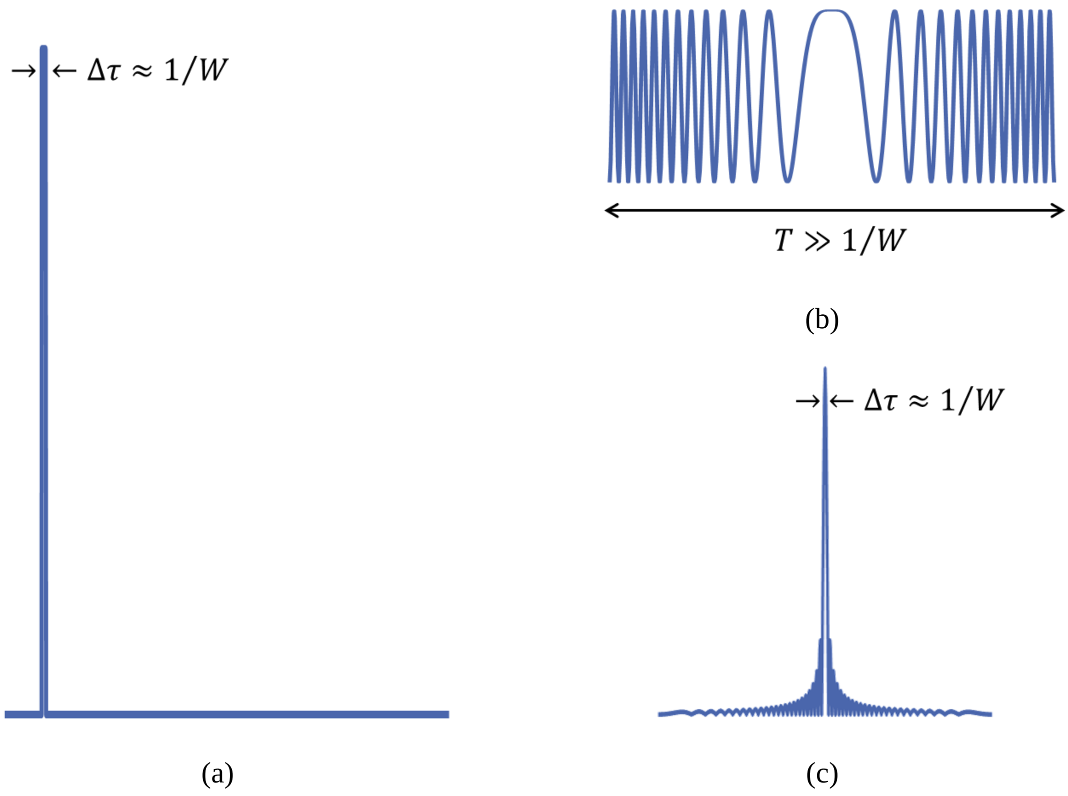


Figure 2.3 – Pulse widths for a high-resolution radar: (a) square pulse; chirped pulse (b) before and (c) after pulse compression.

2.2.1.2 Linear frequency modulation

SAR systems typically employ a linear frequency modulation, also called chirp. The chirped pulse has a constant amplitude during its width T and a frequency that varies linearly over time, sweeping the bandwidth W [1]:

$$f_r(\tau) = f_0 + \zeta\tau, \quad |\tau| \leq T/2 \tag{2.11}$$

$$\zeta = \pm\frac{W}{T} \tag{2.12}$$

The resulting phase is expressed as a quadratic function of time [1]:

$$\varphi_r(\tau) = 2\pi \int f_r(\tau)\, d\tau = \pi\zeta\tau^2, \quad |\tau| \leq T/2 \tag{2.13}$$

Finally, the chirped pulse waveform is given by [1]:

$$s_r(\tau) = A_r\, e^{j\pi\zeta\tau^2}, \quad |\tau| \leq T/2 \tag{2.14}$$

where A_r is the pulse amplitude. Figure 2.3(b) shows only the real part of $s_r(\tau)$ with $f_0 = 0$.

2.2.2 Range Compression

Range compression is the first SAR processing step. It is a matched filter operation that uses the chirped pulse as a reference function. So, each line of the raw data matrix is cross-correlated with the chirped pulse (see Figure 2.1). However, because the fast Fourier transform (FFT) and its inverse (IFFT) are fast, as the name implies, it is much more efficient to perform the analogous operation in the frequency domain. Therefore, the spectrum of each line is multiplied by the conjugate of the chirped pulse spectrum [2], [3].

2.2.2.1 Pulse compression

Suppose for simplicity that the amplitude of the chirped pulse defined in (2.14) is $A_r = 1/\sqrt{T}$. So, for large time-bandwidth products (TW), the spectrum of the chirped pulse can be approximated by [1]:

$$S_r(f) \approx \mathrm{rect}(f/W), \quad TW \gg 1 \tag{2.15}$$

$$\mathrm{rect}(f/W) = \begin{cases} 1, & |f| \leq W/2 \\ 0, & |f| > W/2 \end{cases} \tag{2.16}$$

The assumption $TW \gg 1$ is usually valid for SAR systems. Then, the resulting amplitude of the autocorrelation function is [1]:

$$\left|\Psi_{s_r}(\tau)\right| = |(s_r \star s_r)(\tau)| \approx |\mathrm{sinc}(W\tau)|, \quad |\tau| \ll T \tag{2.17}$$

$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \tag{2.18}$$

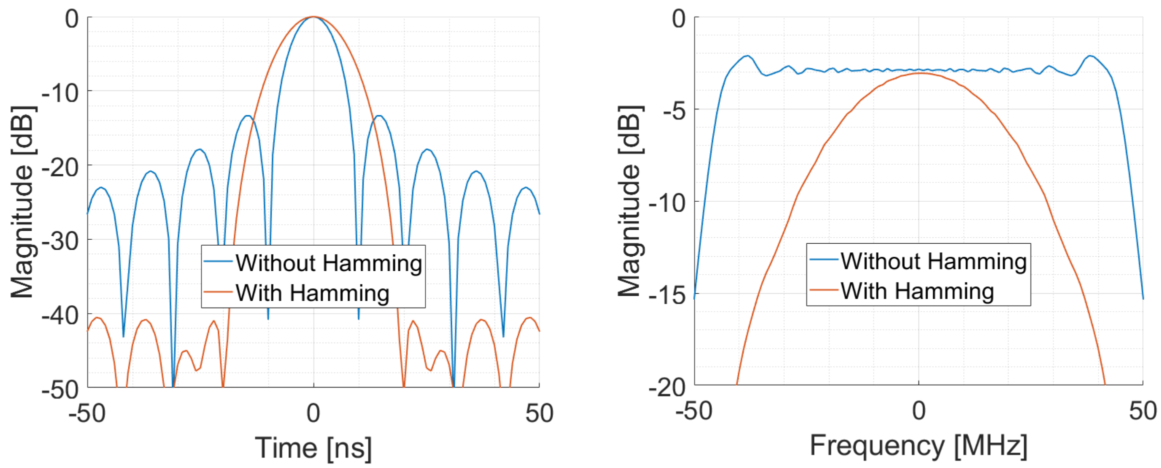
Figure 2.3(c) shows the compressed pulse amplitude $\left|\Psi_{s_r}(\tau)\right|$. As indicated before, the main lobe width is $\Delta\tau \approx 1/W$. Also, the peak-to-sidelobe ratio (PSLR) is 13 dB.

2.2.2.2 Sidelobe reduction

A PSLR of 13 dB is not ideal. Thus, SAR systems often employ a window function to improve the PSLR. One of the most common window functions is the Hamming window with $\xi = 0.54$ [1], [17]:

$$w_H(\tau) = \begin{cases} \xi + (1-\xi)\cos\left(2\pi\dfrac{\tau}{T}\right), & |\tau| \leq T/2 \\ 0, & |\tau| > T/2 \end{cases} \tag{2.19}$$

This window function increases the PSLR to above 40 dB at the cost of widening the main lobe. In Figure 2.4(a), for instance, the width of the main lobe goes from ~10 ns without a window function to ~15 ns with the Hamming window. In addition, the bandwidth of the resulting spectrum is ~40% of the original one, as shown in Figure 2.4(b).



(a) (b)

Figure 2.4 – Compressed pulse with and without Hamming window: (a) time and (b) frequency responses.

2.3 SLOW-TIME AXIS

The azimuth axis of the SAR data matrices (see Figure 2.1) is associated with the platform movement, whose speed is far lower than the speed of light. In addition, the azimuth sampling rate is given by the pulse repetition frequency, whose values range from a few tens to a few thousand Hz, far lower than the range sampling rate. Therefore, the azimuth axis is also called the slow-time axis [2].

2.3.1 Stripmap SAR

Stripmap SAR is the most basic SAR mode, illustrated in Figure 2.5. The platform moves in a linear flight path with constant velocity and altitude. The antenna looks sideways and down at the ground with a fixed pointing vector. The resulting swath width is the ground range extent of the imaged area, whereas the radar signal travels in the slant range (across-track) direction. The flight path direction is called azimuth (along-track).

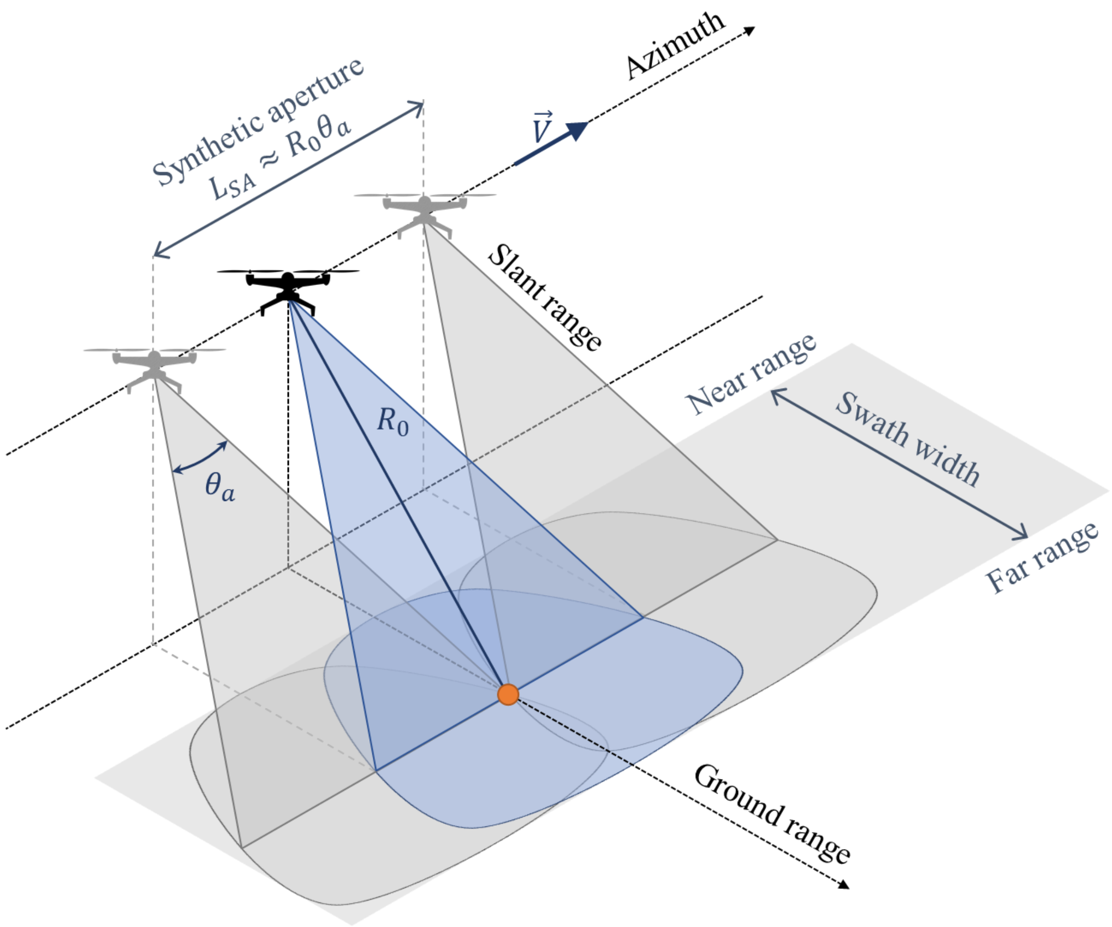


Figure 2.5 – Stripmap SAR imaging.

2.3.1.1 Azimuth resolution

In Figure 2.5, R_0 is the slant range at the point of the closest approach, θ_a is the antenna aperture in azimuth, and L_{SA} is the resulting synthetic aperture length [1]–[3]:

$$L_{SA} \approx R_0 \theta_a \tag{2.20}$$

Note that L_{SA} increases from near-range to far-range while θ_a remains approximately the same.

The corresponding synthetic aperture angle is Θ_{SA} [1]–[3]:

$$\Theta_{SA} = \frac{\lambda}{2L_{SA}} \tag{2.21}$$

where λ is the radar wavelength, and factor 2 comes from the two-way nature of SAR. Therefore, the azimuth resolution is given by [1]–[3]:

$$\delta_a = R_0 \Theta_{SA} = R_0 \frac{\lambda}{2L_{SA}} = \frac{\lambda}{2\theta_a} = \frac{d_a}{2} \tag{2.22}$$

2.3.1.2 Amplitude history

Figure 2.6 shows how the stripmap acquisition geometry translates into the target's amplitude history in the SAR data matrix. The resulting curve in Figure 2.6(b) is hyperbolic.

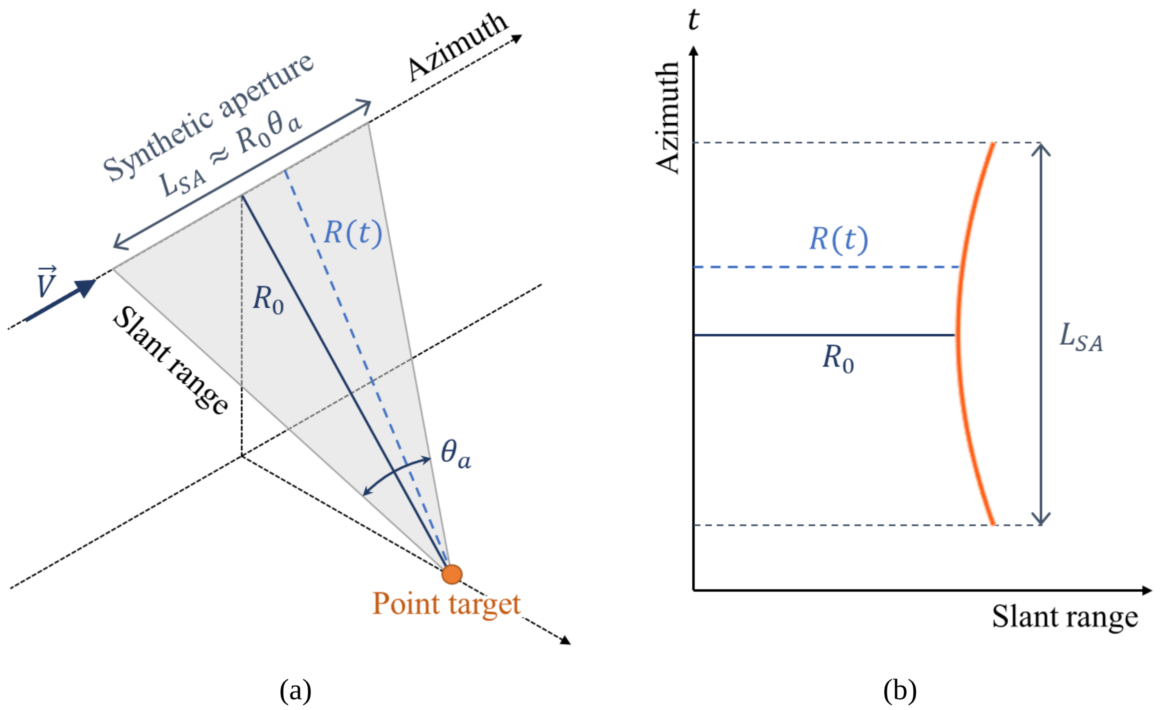


Figure 2.6 –Stripmap SAR amplitude history: (a) acquisition geometry; (b) hyperbole on the data matrix.

In Figure 2.6, the slant range distance varies with the slow-time t. Without loss of generality, consider that the point of closest approach occurs at $t = 0$, i.e., $R(0) = R_0$. Then, the slant range $R(t)$ is given by [2], [3]:

$$R(t) = \sqrt{R_0^2 + (Vt)^2}, \quad |Vt| < L_{SA}/2 \tag{2.23}$$

where V is the platform speed. If $Vt \ll R_0$, then the hyperbole in expression (2.23) can be approximated by a parabole [2], [3]:

$$R(t) \approx R_0 + \frac{(Vt)^2}{2R_0}, \quad Vt \ll R_0 \tag{2.24}$$

The assumption above is usually valid for satellites and aircraft but not so much for drones. Note that the approximation (2.24) is made for simplicity. Accurate SAR processing should consider (2.23) instead [2].

2.3.1.3 Phase history

Because the distance traveled by the radar pulses varies over time, the phase of the backscattered echoes varies accordingly [2], [3]:

$$\varphi_a(t) = -\frac{4\pi}{\lambda}R(t) = \Phi_0 - \frac{4\pi}{\lambda}\frac{V^2}{2R_0}t^2 \tag{2.25}$$

where Φ_0 is constant for a given target. Note how the phase exhibits parabolic behavior (see Figure 2.7(a)).

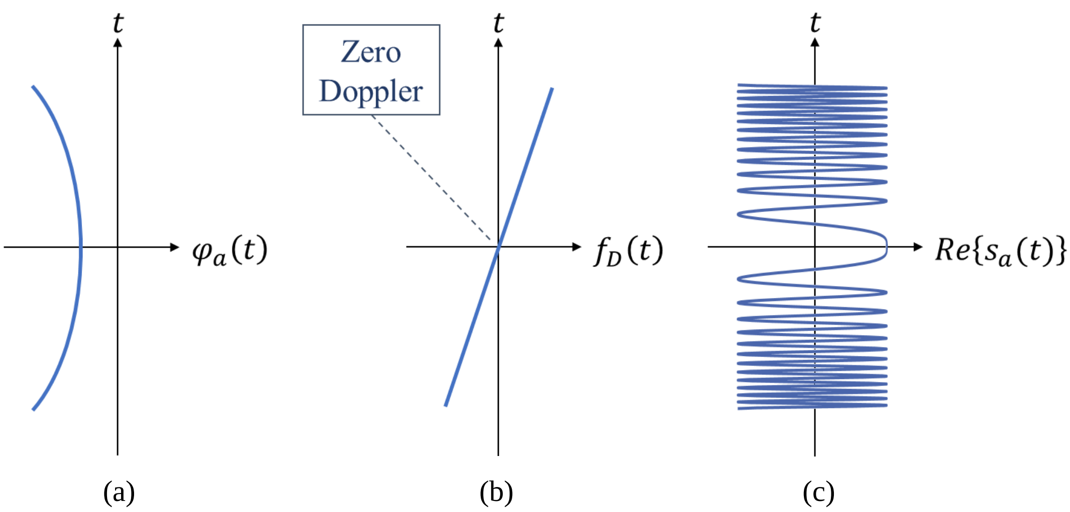


Figure 2.7 –Stripmap SAR phase history: (a) phase, (b) Doppler frequency, and (c) resulting modulated signal against slow-time.

Since the signal frequency is the derivative of the signal phase, it varies linearly with time (see Figure 2.7 (b)) [2]:

$$f_D(t) = \frac{1}{2\pi}\frac{\partial}{\partial t}\varphi_a(t) = -\frac{2V^2}{\lambda R_0}t \tag{2.26}$$

Interestingly, the azimuth frequency turns out to be similar to that of a linear frequency modulated signal. Thus, the phase history takes the form of an azimuth chirp (see Figure 2.7(c)). Furthermore, the azimuth frequency is also called Doppler frequency, and the zero Doppler corresponds to the point of closest approach.

2.3.2 Spotlight SAR

Figure 2.8 shows another common SAR mode, called Spotlight SAR. Again, the platform moves in a linear flight path with constant velocity and altitude. However, the antenna pointing vector is steered in azimuth, so the radar keeps illuminating the same area. As a result, the imaged area is smaller, but the synthetic aperture is longer than Stripmap SAR, providing a finer resolution.

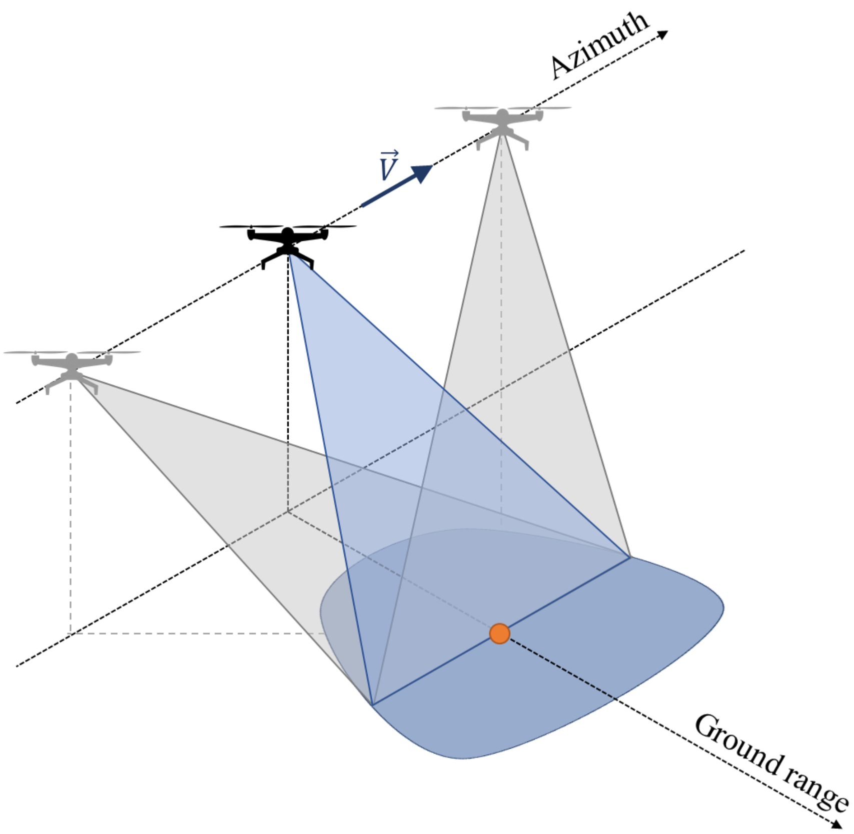


Figure 2.8 – Spotlight SAR imaging.

2.3.3 Frequency-Domain Azimuth Compression

Frequency-domain algorithms perform azimuth compression through a matched filter operation, using the amplitude and phase history as a reference function (see Figure 2.1). Note that the synthetic aperture length increases from near-range to far-range; thus, the reference function must be adjusted accordingly. Similar to range compression, instead of cross-correlating in time, these algorithms conjugate the spectrum of the reference function and multiply in the frequency domain. This procedure is quite effective thanks to the FFT and IFFT [2], [3]. The following subsections shall present a few aspects of frequency-domain algorithms.

2.3.3.1 Flight path dependence

As detailed in Subsection 2.3.1, the azimuth reference function depends on the acquisition geometry. In other words, different acquisition geometries will have different reference functions. For example, even Stripmap and Spotlight SAR would not have identical azimuth and phase histories, which can be noticed by comparing Figure 2.5 and Figure 2.8. Therefore, frequency-domain algorithms depend heavily on the flight path [2], [3].

2.3.3.2 Motion compensation

In Subsection 2.3.1, we assumed that the platform moved with constant velocity and altitude to determine the azimuth reference function. However, that will not hold for actual SAR systems. Consequently, the image data might be defocused, and thus a motion compensation procedure may be required. Motion compensation measures, records, and compensates for the deviations between the real and ideal flight paths [1], [3].

2.3.3.3 Range migration

From equation (2.24), the difference between the maximum and minimum slant ranges will be equal to [1], [2]:

$$\Delta R = \frac{(Vt)^2}{2R_0} = \frac{{L_{SA}}^2}{8R_0} \tag{2.27}$$

If ΔR is greater than the range sampling interval, the amplitude response will occupy more than one range bin along its curvature. This issue is called range migration and needs to be addressed. Fortunately, there are well-established algorithms for that: wavenumber domain or $\omega - k$ [18], range-Doppler [19], [20], and chirp scaling [21], [22] algorithms.

2.3.3.4 Autofocus

Autofocus techniques take information from the image data to estimate and correct phase errors to provide a sharper image. The phase errors have many sources: uncompensated or unmeasured motion, atmospheric propagation effects, hardware limitations, and processing approximations. All these factors can deteriorate the image focus. Standard cost functions for optimizing focus in SAR applications are maximum contrast and minimum entropy [1], [23].

2.3.3.5 Ground range vs. slant range

Frequency-domain algorithms produce the image data in a slant range geometry, though the imaged scene is in ground range geometry. Therefore, the pixels need to be projected onto the ground plane for comparing the image data with maps, for instance. The ground range spacing (δ_{gr}) relates to the slant range spacing (δ_{sr}) by [1]:

$$\delta_{gr} \approx \frac{\delta_{sr}}{\cos(\gamma)} \tag{2.28}$$

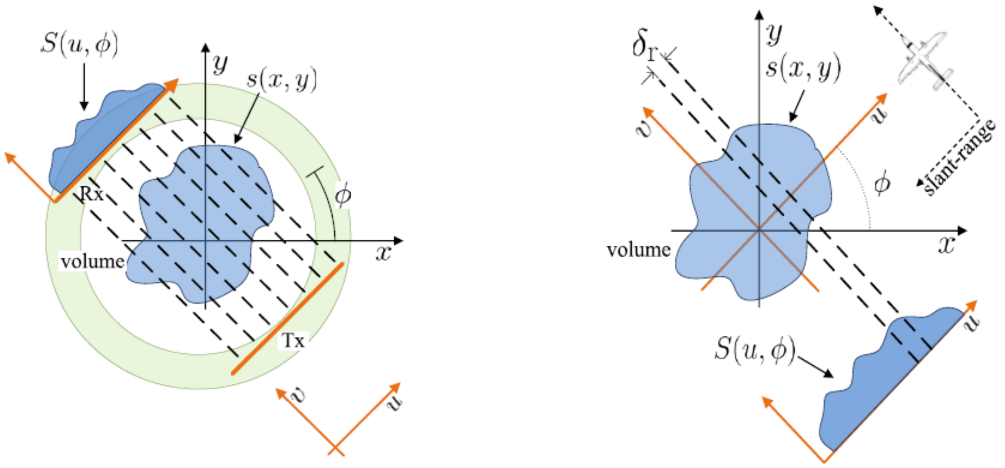
where γ is the grazing angle, which changes from near-range to far-range. Since the slant range spacing is typically constant (and slightly shorter than the range resolution), the ground range spacing will vary across the image data. Therefore, interpolation and resampling are necessary to provide an image in ground range geometry with regular spacing.

2.3.3.6 Geocoding

SAR geocoding references the image data into a geographic map projection, with each pixel corresponding to a latitude and longitude, or northing and easting coordinates. The purpose of geocoding is to describe the data in a common coordinate system so that it can be compared to data from different SAR acquisitions or even from different sensors. Proper geocoding requires a digital elevation model (DEM), which may be obtained with the SAR system by using interferometry [3], [24].

2.4 TIME-DOMAIN BACK PROJECTION

In 1983, Munson *et al.* [25] demonstrated that Spotlight SAR could be interpreted as a tomographic reconstruction problem. Computed tomography (CT) is a technique that provides high-resolution medical images. A 2D tomographic image is a cross-section of a 3D object obtained by processing data from multiple X-ray measurements taken at different observation angles. Each X-ray measurement is a 1D projection, as illustrated in Figure 2.9(a). Something similar occurs in SAR, with a stronger analogy in the spotlight mode. For each observation angle, the radar takes several samples (1D projection) of the backscattered echoes coming from the imaged area (see Figure 2.9(b)). Munson *et al.* [25] also showed that an algorithm developed for CT could be adapted for SAR. That is how the time-domain BP algorithm originated.



(a) (b)

Figure 2.9 – Analogy between (a) computed tomography (CT) and (b) Spotlight SAR; $s(x, y)$ is the unknown 2D image, $S(u, \phi)$ is the 1D projection onto the u-axis, which rotates for each observation angle ϕ. Source: Ponce *et al.* [26] © 2016 IEEE.

2.4.1 Basic Algorithm

The first key difference in the BP algorithm is that the information taken from the range-compressed data is processed onto an image sample grid. Figure 2.10 shows the steps of the BP algorithm. Step 2 is not required if the radar constantly illuminates the imaged area (e.g., Spotlight SAR). Furthermore, if step 2 is necessary (e.g., Stripmap SAR), it could be executed for image blocks instead of pixel by pixel [17]. Then, steps 3 to 7 must be repeated for each azimuth position that belongs to the synthetic aperture and for every pixel.

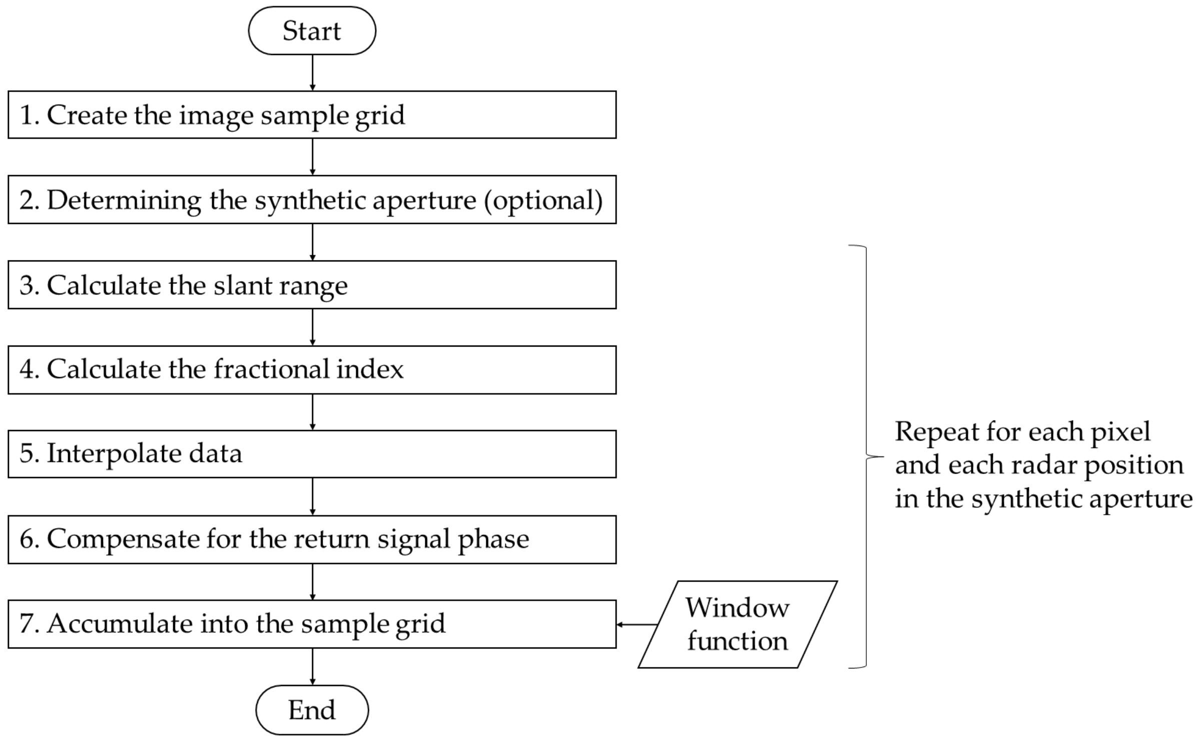


Figure 2.10 – Flowchart of the Back Projection (BP) algorithm. Source: Modified from Doerry *et al.*, 2016 [27].

2.4.1.1 Create the image sample grid

To create the image sample grid, first, sample the area of interest so that each pixel corresponds to a position in space. Next, choose an appropriate sampling spacing to avoid aliasing, considering the flight path geometry and the original sampling rates in range and azimuth [17]. Finally, retrieve the height of each pixel using a DEM. Thankfully, data from the Shuttle Radar Topography Mission (SRTM) is openly available, covering over 80 % of the Earth's land surface with a one arc-second sampling spacing (around 30 m) [28].

This procedure might require transformations between different coordinate systems. SRTM data is provided in geodetic coordinates. The area of interest could be sampled in geodetic coordinates or a local coordinate system, such as East-North-Up (ENU) or North-East-Down (NED). The final image sample grid must be represented in the same coordinate system as the navigation data to allow calculations in the following steps. In addition, both must be represented in a cartesian coordinate system, which could be either a local system (e.g., ENU or NED) or the Earth-centered Earth-fixed (ECEF) system. Figure 2.11 shows the image sample grid and the radar positions in an ENU system.

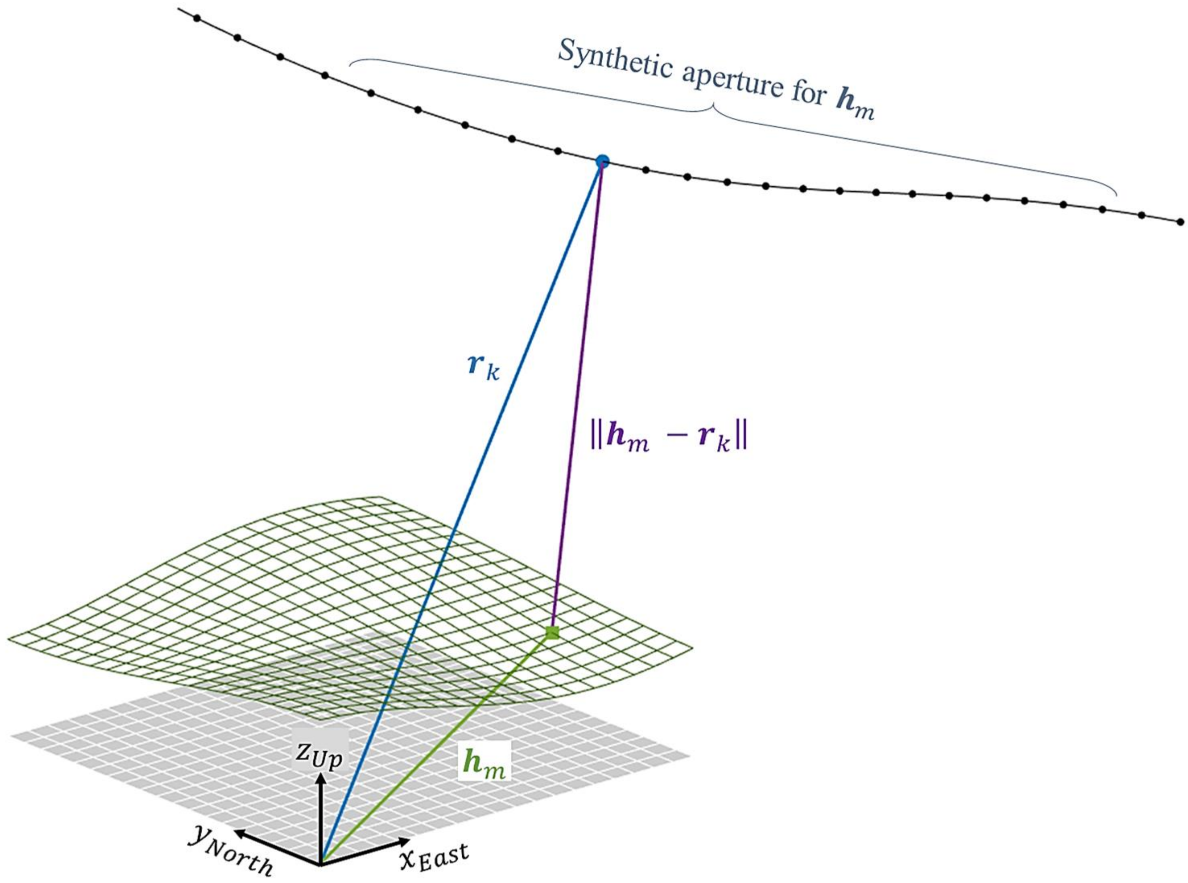


Figure 2.11 – Schematic view of the reconstruction geometry for processing SAR data from a nonlinear flight path with the BP algorithm. Source: Modified from Frey *et al*. [17] © 2009 IEEE.

2.4.1.2 Determine the synthetic aperture (optional)

Determining the synthetic aperture is finding the radar positions ($\mathbb{K}_m$) for which the antenna is illuminating the pixel or image block. There are different methods to accomplish that. For instance, Frey *et al*. [17] use the antenna pointing vector and the Doppler frequency to indirectly measure the angle between the slant range and the antenna axis. However, as said before, this step might not be required.

2.4.1.3 Calculate the slant range

The slant range ($R_{m,k}$) is given by [17], [27]:

$$R_{m,k} = \|\boldsymbol{h}_m - \boldsymbol{r}_k\| \tag{2.29}$$

where $\boldsymbol{r}_k$ is the position vector of the k^{th} radar position and $\boldsymbol{h}_m$ is the position vector associated with the m^{th} pixel (see Figure 2.11).

2.4.1.4 Calculate the fractional range index

The radar transmits a pulse, and after a buffer delay, it starts sampling the return signal. Thus, the slant range $R_{m,k}$ corresponds to a specific range bin sample in the k^{th} azimuth line of the range-compressed data. To find the correct range bin, we need to calculate the fractional range index [27]:

$$\nu_{m,k} = \frac{R_{m,k} - R_{delay}}{\delta_{sr}} \tag{2.30}$$

where δ_{sr} is the range sampling interval, and R_{delay} is the slant range of the first range bin associated with the buffer delay. Note that $\nu_{m,k}$ is not an integer number.

2.4.1.5 Interpolate data

The fractional range index $\nu_{m,k}$ often falls between two range bin samples. Therefore, interpolation is required to find the equivalent value for $g(k, \nu_{m,k})$, where g is the range-compressed data. Doerry *et al.* [29] propose a design methodology for an interpolation filter based on the following criteria: limiting the aliased energy into the bandwidth of interest to -40 dB. Then, each interpolation method will require a different fractional bandwidth (bw) to comply with the aliasing criteria:

- The nearest neighbor interpolation requires $bw = 0.02$;
- The linear interpolation requires $bw = 0.184$;
- The cubic B-spline requires $bw = 0.264$.

Therefore, if the range-compressed data has $bw = 0.9$, it will need to be up-sampled by a factor of: 0.9/0.02 = 45 for the nearest neighbor interpolation; 0.9/0.184 = 4.89 for the linear interpolation; 0.9/0.264 = 3.41 for the cubic B-spline interpolation. Up-sampling can be applied by zero-padding the data spectrum.

2.4.1.6 Compensate for the return signal phase

The data $g(k, \nu_{m,k})$ of pixel $\boldsymbol{h}_m$ will have a different phase for each radar position $\boldsymbol{r}_k$. Therefore, we need to compensate for the phase differences so that all contributions are coherently combined [27]:

$$\tilde{g}(k, \nu_{m,k}) = g(k, \nu_{m,k})e^{j\varphi_{m,k}} \tag{2.31}$$

$$\varphi_{m,k} = +\frac{4\pi}{\lambda} R_{m,k} \tag{2.32}$$

where λ is the radar wavelength. Note that the phase compensation term $\varphi_{m,k}$ has a positive sign for balancing the phase delay of the return signal in expression (2.5).

2.4.1.7 Accumulate into the sample grid

The last step is to accumulate the compensated data into the image grid. Finally, when steps 3 to 7 have been repeated for every radar position within the synthetic aperture ($k \in \mathbb{K}_m$), the image datum for the pixel $\boldsymbol{h}_m$ becomes [27]:

$$s(\boldsymbol{h}_m) = \sum_{k \in \mathbb{K}_m} w_{m,k}\, g(k, \nu_{m,k})e^{j\varphi_{m,k}} \tag{2.33}$$

The window function $w_{m,k}$ can be chosen at will. For instance, Frey *et al*. [17] use a Hamming window function with the Doppler frequency in its argument in such a way that $w_{m,k} = 1$ when the antenna axis is pointing at the pixel. On the other hand, the window function could be as simple as $w_{m,k} = 1$ for all radar positions. Figure 2.12 summarizes the BP algorithm.

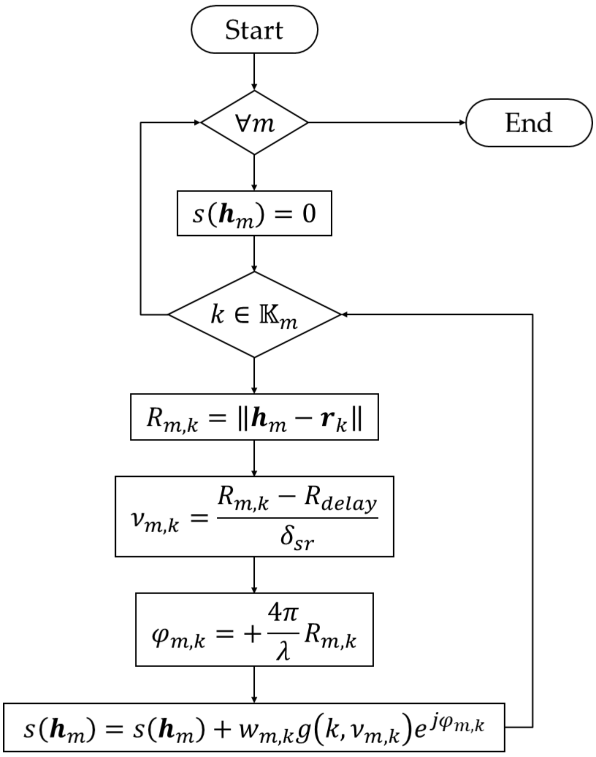


Figure 2.12 – Flowchart of the BP algorithm with equations.

2.4.2 Comparison to Frequency-Domain Algorithms

This subsection compares the BP algorithm to the frequency-domain algorithms of Subsection 2.3.3. Table 2.2 presents an overview, and the next paragraphs give further details.

Table 2.2 – Comparison overview between azimuth compression algorithms.

Frequency-Domain Algorithms	Time-Domain BP Algorithm
☒ Work for specific flight paths	☑ Works with arbitrary flight paths
☒ May need motion compensation	☑ Does not need motion compensation
☒ Need an additional step to represent the image data in ground range geometry	☑ The image data is already represented in ground range geometry
☒ Need an additional step to create a geocoded image data	☑ The image data is already geocoded
☒ May need autofocus	☒ May need autofocus
☑ Do not need previous knowledge of the terrain topography	☒ Needs previous accurate knowledge of the terrain topography
☑ Low computing cost	☒ High computing cost

2.4.2.1 Flight path dependence

Frequency-domain algorithms depend highly on the flight path since an azimuth reference function is required to execute the azimuth compression. In contrast, The BP algorithm does not require an azimuth reference function, thus working for arbitrary flight path geometries. This characteristic is the main advantage of the BP algorithm.

2.4.2.2 Computing costs

Frequency-domain algorithms are cost-efficient, thanks to employing the FFT and the IFFT. In reverse, the BP algorithm has a high computing cost. The overall computing time can be somewhat reduced by parallelizing the data processing. Still, the high computing cost is the main disadvantage of the BP algorithm.

Fortunately, another time-domain algorithm, called fast factorized back projection (FFBP), uses a divide-and-conquer strategy to reduce the computing cost [14], [30], [31]. Hence, a 3D FFBP algorithm is one of the contributions of this thesis. Incidentally, a divide-and-conquer strategy is also the reason behind the efficiency of the FFT and IFFT [32].

2.4.2.3 Other aspects

Since the actual flight path is non-ideal, frequency-domain algorithms may require motion compensation. However, this procedure is unnecessary for the BP algorithm because it already uses the actual radar positions to perform azimuth compression. On the other hand, phase errors may defocus the image data, so both frequency- and time-domain algorithms may require autofocus to correct those errors. In the case of BP algorithms, phase errors can be caused by inaccuracies in the navigation data or the DEM [23].

Frequency domain algorithms need an additional step for representing the image data in ground range geometry and another for geocoding. In contrast, neither of these steps is required in the BP algorithm, thanks to the image sample grid. The only inconvenience is that the BP algorithm calls for previous knowledge of the terrain topography, which can be partially overcome thanks to SRTM data.

3 FAST FACTORIZED BACK PROJECTION*

SAR image processing requires efficient algorithms in terms of accuracy and processing time. Frequency-domain algorithms are fast but perform better when the flight path is linear and free of motion errors. The time-domain BP algorithm can process SAR data for any flight path with high focusing quality but at high computing costs. FFBP algorithms can significantly reduce the processing time while maintaining the BP algorithm’s accuracy. However, the increased sophistication makes it challenging to formulate an FFBP algorithm suitable for any flight path. As a result, many FFBP algorithms either assume a linear flight path to simplify calculations [30], [31], [33], [34] or are tailored for circular flight paths [35], [36].

This chapter proposes an FFBP algorithm with two key original features. First, unlike other FFBP algorithms, it is suitable for any flight path geometry, thanks to a data mapping approach. The only assumption is that the radar constantly illuminates the imaged area or volume. Second, it employs a flexible tree structure that can easily handle 2D and 3D data. This tree structure is a modified version of a classical tree structure.

To the best of the author’s knowledge, the proposed FFBP algorithm is the first one capable of processing nonlinear SAR data in full-3D. Full-3D processing means that the volumetric image is not sliced into several 2D layers to be processed separately. Another work [37] presented an FFBP algorithm that could process images in full-3D. However, it was designed for downward-looking sonar systems moving in quasi-linear paths.

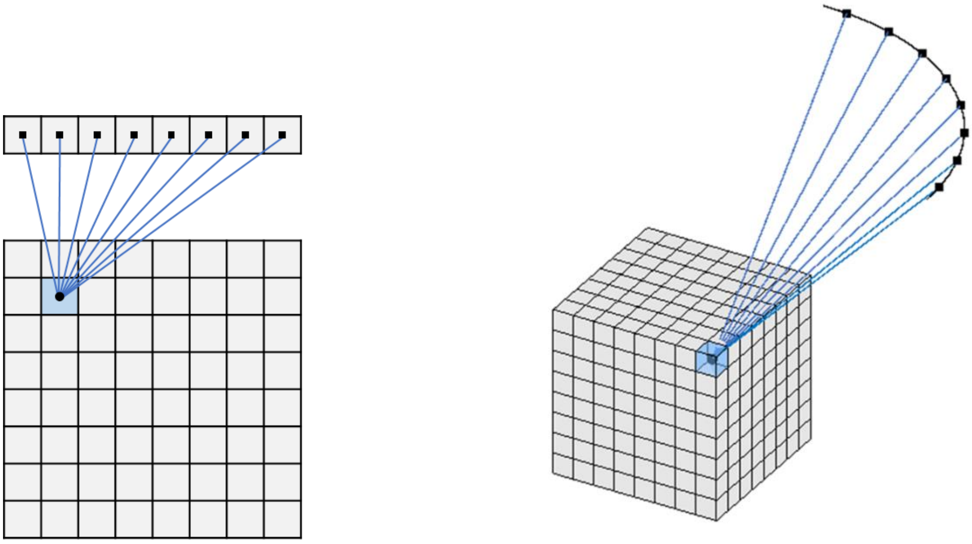
The proposed FFBP algorithm is much faster than the BP algorithm. It achieves speed-up factors of up to 21 for 3D and 13 for 2D images while obtaining low phase errors and high degrees of coherence.

Section 3.1 explains the main strategies behind FFBP algorithms. Then, Section 3.2 describes the proposed FFBP algorithm. Next, Section 3.3 presents a proof of concept using simulation results. Finally, Section 3.4 performs a phase error analysis using actual SAR data.

* This chapter comprises revised and expanded material from two publications by the author. Sections 3.2 and 3.3 are derived from [13] © 2020 IEEE. Sections 3.2, 3.3.4 and 3.4 are derived from [14] licensed under CC BY 4.0. New details were added as text, equations, figures, tables and even subsections.

3.1 PROCESSING TIME REDUCTION

As seen before, the BP algorithm can produce well-focused images for arbitrary flight paths but at high computing costs. Such limitation is because the BP algorithm integrates the information from all SAR pulses in one go for each pixel or voxel. Thus, if the synthetic aperture has P pulses and the output image has P^2 pixels or P^3 voxels, the number of operations is $\mathcal{O}(P^3)$ for the 2D image and $\mathcal{O}(P^4)$ for the 3D image. Figure 3.1 illustrates this for $P = 8$.



(a) (b)

Figure 3.1 – The BP algorithm: (a) integrating one pixel of a 2D image; (b) integrating one voxel of a 3D image.

There are two main strategies for reducing processing time commonly seen in the literature: the divide-and-conquer strategy, first proposed in 1996 by McCorkle and Rofheart [30]; and representing data in polar coordinates, first proposed in 1999 by Yegulalp [38]. In 2003, Ulander *et al*. [31] combined both strategies to reduce further the processing time; they also provided a means of controlling the phase error. More recent strategies include, for instance: processing range data as a bulk [39]; applying spectrum compressing filters to reduce the Nyquist rate [33].

The divide-and-conquer strategy is at the core of most FFBP algorithms. It shall be explained in Subsection 3.1.1. Many FFBP algorithms operate in polar coordinates. However, some authors still prefer to work with cartesian coordinates. The advantages and disadvantages of using polar coordinates shall be presented in Subsection 3.1.2. Then, Subsection 3.1.3 briefly discusses the error analysis performed by Ulander *et al*. [31].

3.1.1 Divide-and-Conquer

Chapter 2 shows that each radar pulse has a poor resolution in azimuth and that high resolution can be achieved by integrating the synthetic aperture. McCorkle and Rofheart [30] proposed making this integration in recursive steps: at each recursion, subapertures are merged, becoming progressively longer, whereas subimages are split, becoming increasingly smaller. This process constitutes the divide-and-conquer strategy. It is illustrated in Figure 3.2 and Figure 3.3 for 2D and 3D images, respectively.

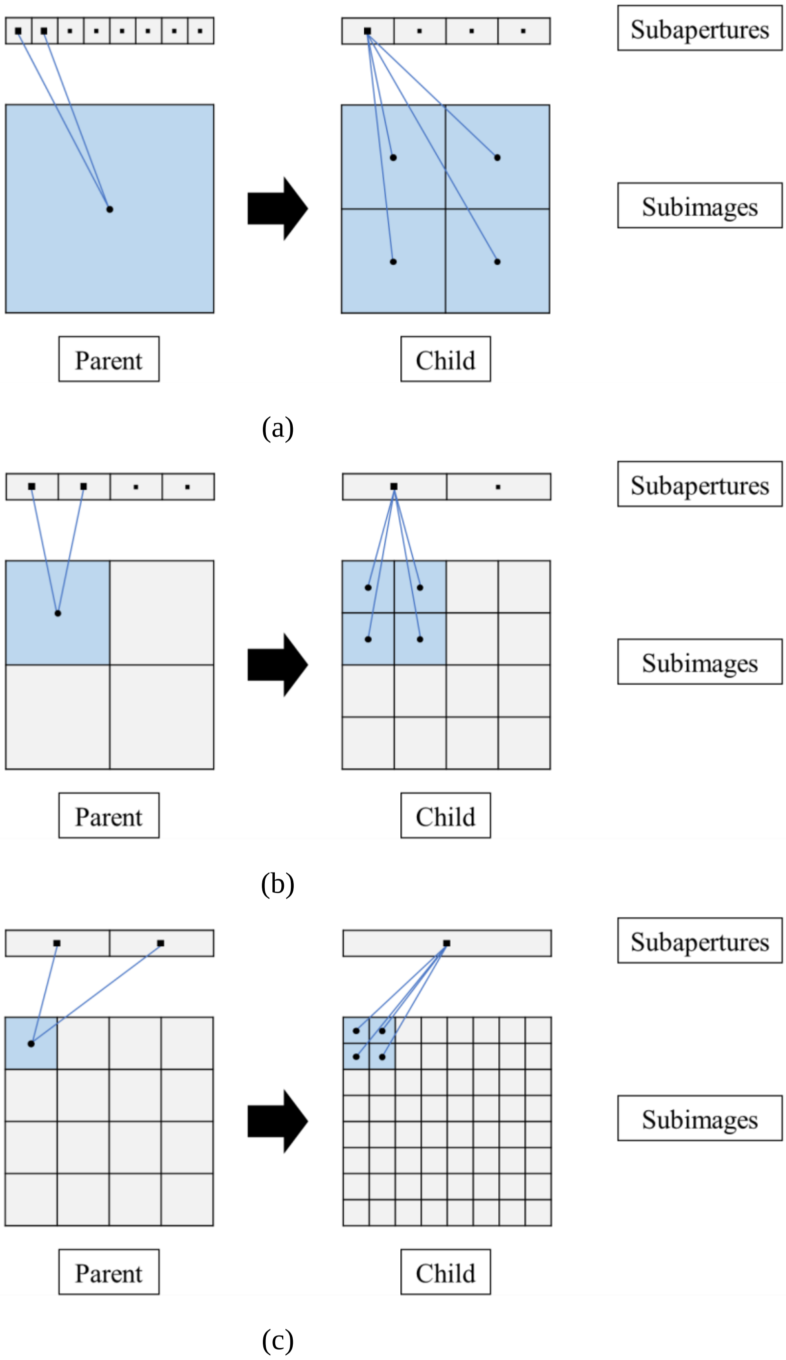
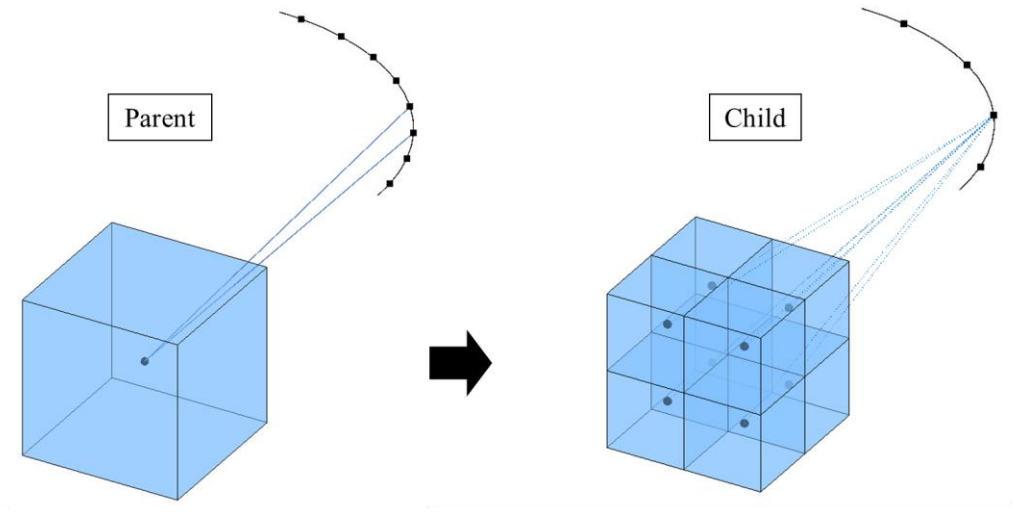
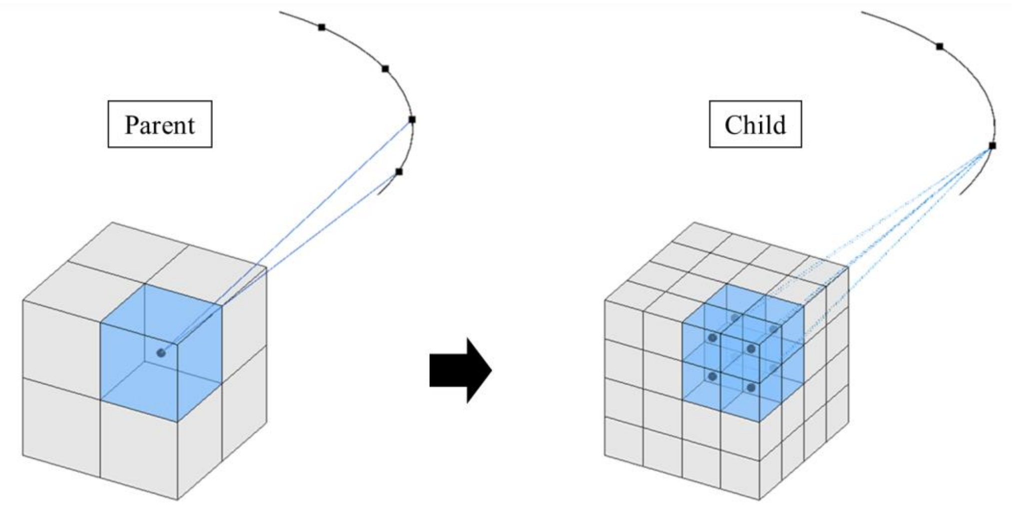


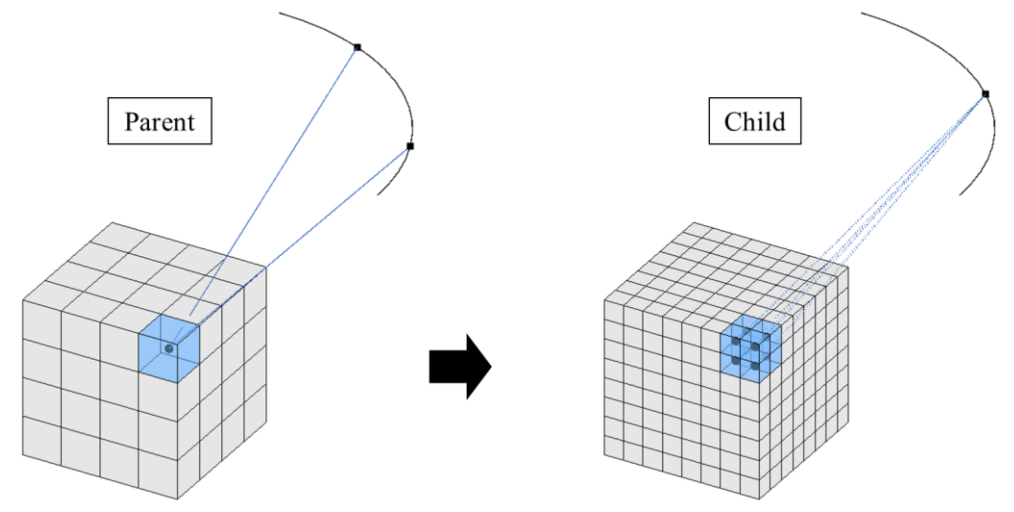
Figure 3.2 – The divide-and-conquer strategy for a 2D image highlighting the parent-child dynamic: (a) the first recursion; (b) the second recursion; (c) the final recursion.



(a)



(b)



(c)

Figure 3.3 – The divide-and-conquer strategy for a 3D image highlighting the parent-child dynamic: (a) the first recursion; (b) the second recursion; (c) the final recursion.

McCorkle and Rofheart [30] also proposed organizing the data into a tree structure. First, the root node represents all the original data, i.e., the inputs to the algorithm. Then, since the data are processed recursively, they are organized in a parent-child dynamic. The parent node represents all the data from the previous recursion; it behaves like the input SAR data of the BP algorithm. The child node represents the data processed in the current recursion. At the end of each recursion, children become parents. Figure 3.2 and Figure 3.3 also highlight the parent-child dynamic.

3.1.1.1 Range samples

Figure 3.4 shows another feature of McCorkle and Rofheart's algorithm [30]. At each recursion, range samples are taken along the lines connecting the centers of each child subaperture and each child subimage. Since the dimensions of the child subimage decrease with each recursion, the number of range samples also decreases. When the desired resolution is reached, only one range sample remains per subimage (e.g., the right side of Figure 3.2(c) and Figure 3.3(c)).

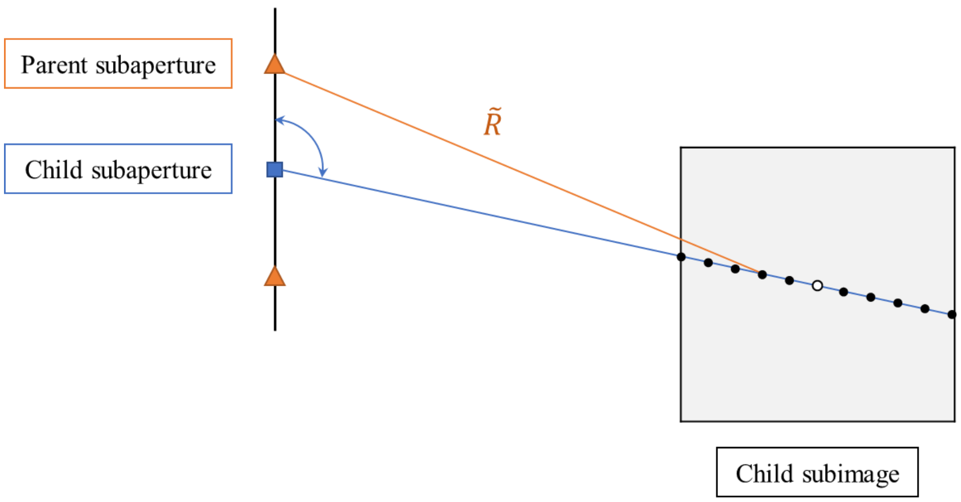


Figure 3.4 – The range samples taken at a child subimage and the triangle formed by a range sample, a child subaperture, and a parent subaperture. The slant range $\tilde{R}$ is estimated using the law of cosines. Source: Modified from McCorkle and Rofheart [30] © 1996 SPIE.

Drawing a parallel with the BP algorithm, the parent range samples behave like the range bin samples of the SAR data, whereas the child range samples behave like the pixels of the processed image. Moreover, computing the child data requires estimating the slant range $\tilde{R}$ between each child range sample and each parent subaperture. These two points and the child subaperture form a triangle (see Figure 3.4), for which it is easy to determine two sides and one angle. Therefore, the slant range $\tilde{R}$ is estimated using the law of cosines.

3.1.1.2 Number of operations

The divide-and-conquer strategy reduces processing time by lowering the number of operations. Let L be the number of subapertures that are combined at each recursion. If the synthetic aperture has $P = L^N$ pulses and the image has $P^2 = L^{2N}$ pixels, then the FFBP algorithm will require $N = \log_L P$ recursions.

At the n^{th} recursion, the number of operations depends on:

- The number of parent subapertures merged into each child subaperture (L);
- The number of child subapertures (Q_n^{sa});
- The number of child subimages (Q_n^{si});
- The number of range samples taken at each child subimage (M_n).

The number of child subapertures reduces by L at each recursion [30]:

$$Q_n^{sa} = \frac{P}{L^n} \tag{3.1}$$

In contrast, the number of child subimages increases by L^2 at each recursion [30]:

$$Q_n^{si} = L^{2n} \tag{3.2}$$

The number of range samples is proportional to the diagonal of the child subimage, which decreases by L at each recursion. Therefore, the number of range samples is [30]:

$$M_n \propto \frac{P}{L^n} \tag{3.3}$$

Consequently, the total number of operations is of the order [30]:

$$\begin{aligned} \mathcal{O}\left(\sum_{n=1}^{N} L \times Q_n^{sa} \times Q_n^{si} \times M_n\right) &= \mathcal{O}\left(\sum_{n=1}^{N} L \times \frac{P}{L^n} \times L^{2n} \times \frac{P}{L^n}\right) = \mathcal{O}\left(\sum_{n=1}^{N} LP^2\right) \\ &= \mathcal{O}(LP^2 N) = \mathcal{O}(LP^2 \log_L P) = \mathcal{O}(P^2 \log P) \end{aligned} \tag{3.4}$$

Since these are asymptotical approximations, we can neglect the constant L and the logarithm base. As a result, the divide-and-conquer strategy can provide a speed-up factor of $P/\log P$ for 2D images.

For 3D images, $Q_n^{si} = L^{3n}$. Thus, expression (3.4) becomes:

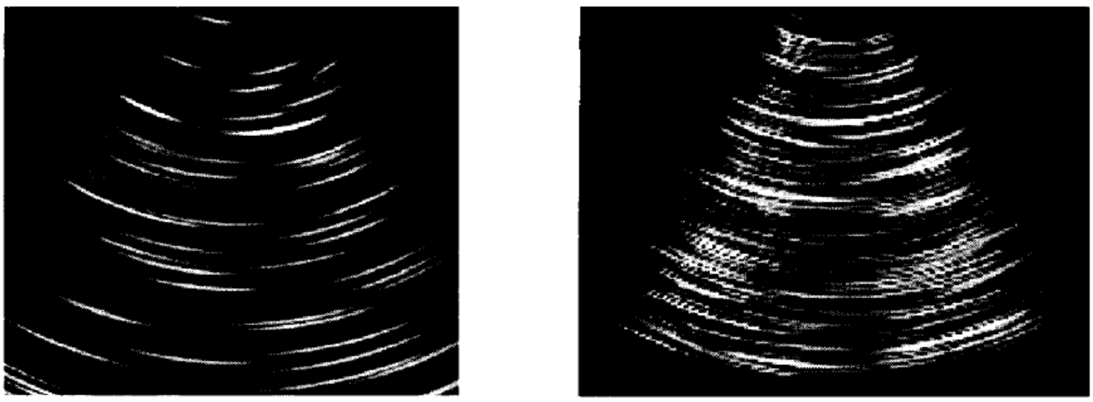
$$\mathcal{O}\left(LP^2\sum_{n=1}^{N}L^n\right) = \mathcal{O}\left(LP^2\frac{L(L^N-1)}{L-1}\right) = \mathcal{O}\left(\frac{L^2}{L-1}P^2(P-1)\right) = \mathcal{O}(P^3) \tag{3.5}$$

Therefore, for 3D images, the theoretical speed-up factor is P rather than $P/\log P$. An important implication is that processing an image in full-3D would be faster than processing its 2D layers separately.

3.1.2 Polar Grids

Yegulalp [38] proposed dividing the synthetic aperture into subapertures only once, i.e., not recursively, and then processing each subaperture with the BP algorithm. The key to reducing processing time is to use a coarse image grid for each subaperture. Once all subapertures have been processed, the images are upsampled to a full resolution and finally combined. Moreover, Yegulalp proposed creating the image grids in local polar coordinates to reduce even further the processing time. The reason is that polar grids have far lower azimuth bandwidths than cartesian grids, thus allowing for a coarser sampling.

Figure 3.5(a) shows the image from a small subaperture processed onto a cartesian grid. This image has a high resolution in range but a low resolution in azimuth. Also, it was sampled close to the Nyquist rate, as indicated by its 2D FFT in Figure 3.5(b). If the sampling were even coarser, aliasing would occur.



(a) (b)

Figure 3.5 – Image from a small subaperture processed onto a cartesian grid: (a) processed image and (b) its 2D FFT. Source: Yegulalp [38] © 1999 IEEE.

In contrast, Figure 3.6(a) shows the same data as Figure 3.5(a), but it was processed onto a polar grid. The azimuth bandwidth is narrow since the targets appear as straight lines rather than arcs, as seen in Figure 3.6(b). A narrow bandwidth means the image is oversampled; thus, the sampling could be much coarser.

(a) (b)

Figure 3.6 – Image from a small subaperture processed onto a polar grid: (a) processed image and (b) its 2D FFT. Source: Yegulalp [38] © 1999 IEEE.

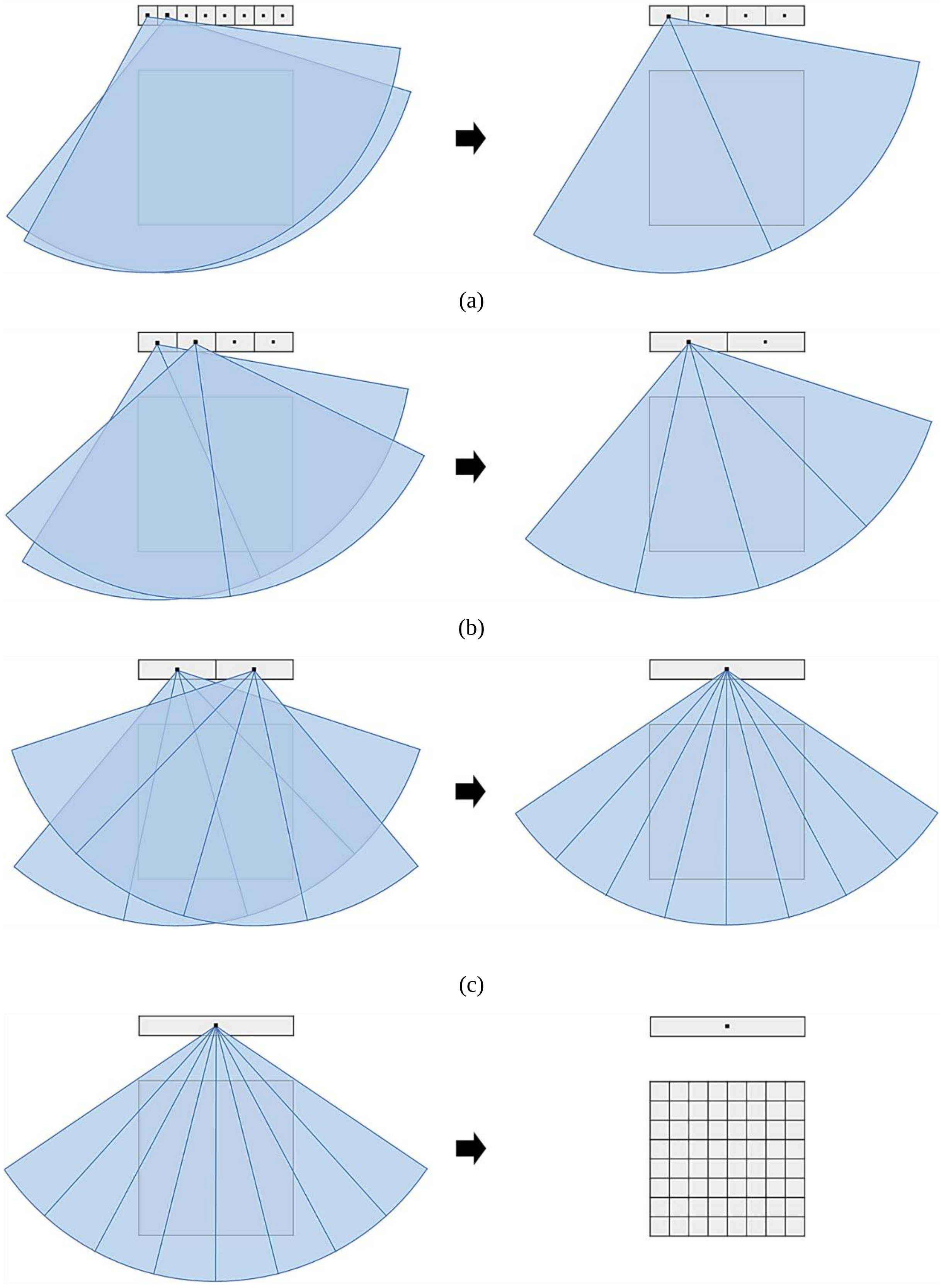
This strategy can also reduce the processing time by lowering the number of operations. For instance, if a 2D image has P^2 pixels and each subaperture has l pulses, the maximum speed-up factor will occur when $l = \sqrt{P}$. Also, if the total number of radar pulses is P, the number of operations is $O(P^{2.5})$.

A speed-up factor of $\sqrt{P}$ is lower than $P/\log P$. Therefore, polar grids are often combined with the divide-and-conquer strategy to produce more efficient algorithms. However, this combination generates some issues.

3.1.2.1 Combining strategies

Ulander *et al.* [31] were the first to propose combining polar grids with the divide-and-conquer strategy. They also coined the name Fast Factorized Back Projection (FFBP). At each recursion, as subapertures are merged, becoming progressively longer, the beams in the polar grid are split, becoming increasingly narrower. After the final recursion, the polar grid is interpolated onto a cartesian grid (see Figure 3.7).

The level of sophistication increases when introducing polar grids to a recursive process. Each polar grid is defined on a local coordinate system centered at the subaperture. Thus, their origins change with each recursion, as seen in Figure 3.7. Therefore, Ulander *et al.* [31] assumed a linear flight path to simplify calculations.



(a) (b) (c) (d)

Figure 3.7 – Combining polar grids with the divide-and-conquer strategy: (a) the first recursion; (b) the second recursion; (c) the final recursion; (d) from polar to cartesian coordinates.

Ponce *et al.* [35] adapted the FFBP to process circular flight paths, approximating them by equilateral polygons. In [26], however, it is implied that this algorithm can only process 2D images. In contrast, Marston *et al.* [37] successfully processed 3D images by defining the image grids in spherical coordinates. However, not only did they assume linear paths, but they also simplified calculations by making each spherical grid point downwards. Unlike what is indicated in Figure 3.7, the axis of each spherical grid did not point to the image center. The reason is that their algorithm was proposed for a downward-looking sonar system.

3.1.2.2 Issues

Using polar grids may be cumbersome [39]. Each polar grid has a different origin, so merging them requires interpolation. The accumulated interpolation errors may result in a loss of accuracy of the final image [33], [34], [36]. In order to mitigate this issue, high-order interpolation kernels could be used, but that could lead to a loss of efficiency. Therefore, other solutions have been proposed in the literature.

For instance, Dong *et al.* [33] apply spectrum compressing filters to reduce the Nyquist rate required for working with cartesian grids. As a result, their algorithm is faster and more accurate than an FFBP using polar grids. However, it only works for quasi-linear flight paths. On the other hand, Sun *et al.* [34] use a global polar coordinate system centered at the final synthetic aperture. Because of that, their algorithm also improves accuracy while reducing processing time. Nevertheless, it still assumes a linear flight path to simplify calculations.

3.1.3 Phase Error

According to McCorkle and Rofheart [30], their algorithm reduced computing costs at the expense of processing errors. However, these errors were controllable and could be balanced against speed in a direct trade-off.

Later, Ulander *et al.* [31] provided a more thorough error analysis. By assuming a linear flight path, they found that the range error ($\Delta\tilde{R}$) is bounded by [31]:

$$\left|\Delta\tilde{R}\right| \leq \begin{cases} \dfrac{\delta_k \Delta_c}{4\tilde{R}}, & \delta_k \leq 2\tilde{R} \\ \dfrac{\delta_k \Delta_c}{2}, & \delta_k > 2\tilde{R} \end{cases} \tag{3.6}$$

where $\tilde{R}$ is the estimated slant range (see Figure 3.4), δ_k is the length of the child subaperture, and Δ_c is the width of the child subimage. In the case of deviations from a linear flight path, the range error would increase, and a different expression would be required.

The maximum range error in each recursion is given by [31]:

$$\left|\Delta\tilde{R}\right| \leq \frac{N_p \delta_p \delta_k^0}{4R_{min}} \tag{3.7}$$

where δ_p is the pixel spacing, N_p is the number of pixels in the along-track direction, δ_k^0 is the aperture spacing, and R_{min} is the closest range from the radar to the image.

Finally, the phase error can be obtained by multiplying the range error by $4\pi/\lambda$. Hence, Ulander *et al*. [31] proposed the following method to keep the phase error below a given threshold:

- Calculate the maximum subimage size for the first recursion step using expression (3.7). Then, divide the image into blocks to be processed separately;
- Compensate the increase in subaperture length with an equivalent decrease in subimage width to keep the range error constant (see expression (3.6)).

3.2 THE ALGORITHM

The proposed FFBP algorithm employs the divide-and-conquer strategy, though it operates in cartesian coordinates. From Section 3.1, we know that cartesian grids make FFBP algorithms slower than polar or spherical grids. However, they are much easier to implement, allowing the design of an FFBP algorithm that processes images in full-3D for any flight path.

The FFBP algorithm is parallelizable, so the image is split into blocks to be processed in parallel (see Figure 3.8(a)). Furthermore, the data are processed recursively in a parent-child dynamic (see Figure 3.8(b)). Moreover, the proposed FFBP algorithm is vectorized. So, in the following subsections, matrix indices shall be written within parentheses to distinguish them from for-loop indices, written as subscripts. Also, variables representing positions in the (x, y, z) space shall be written in bold letters.

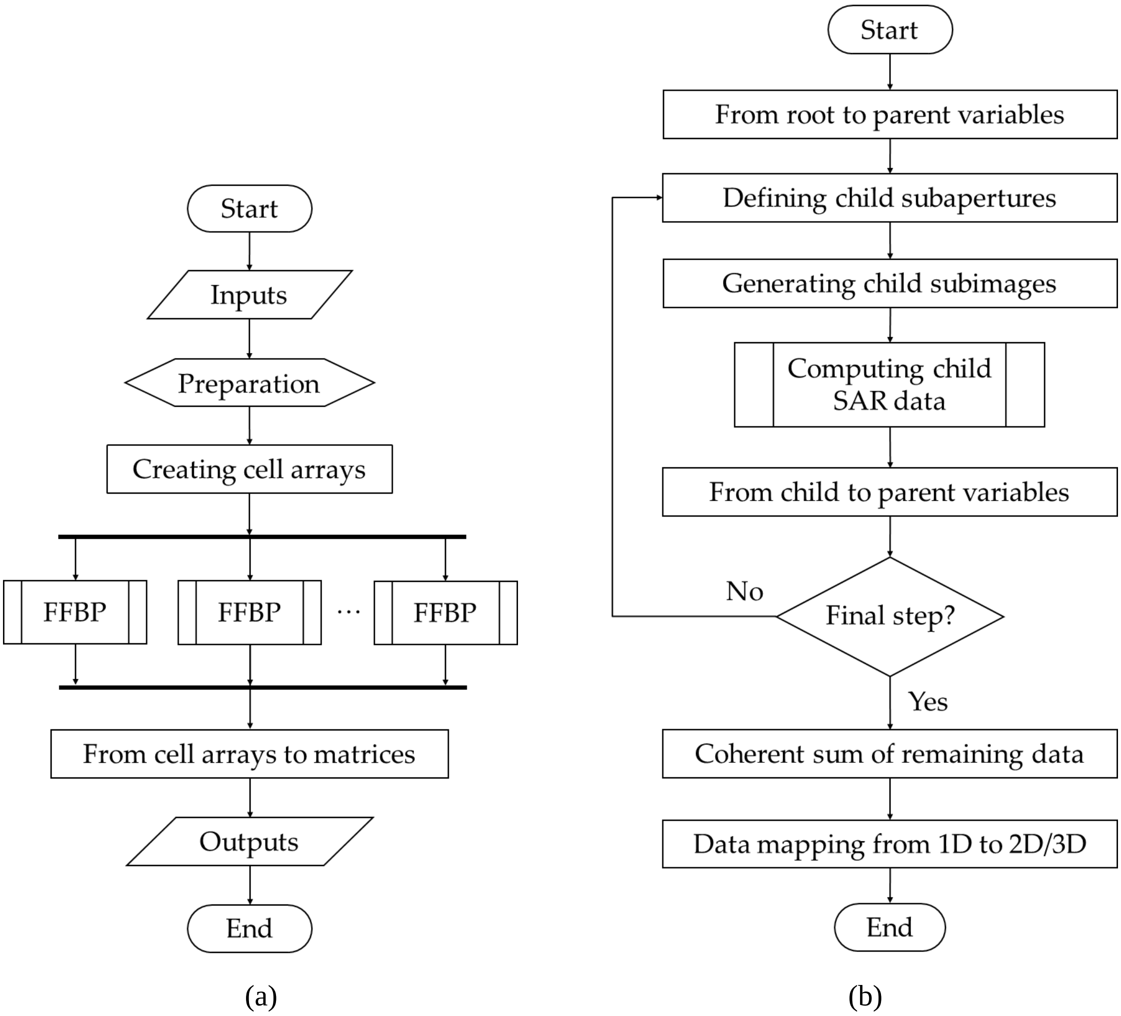
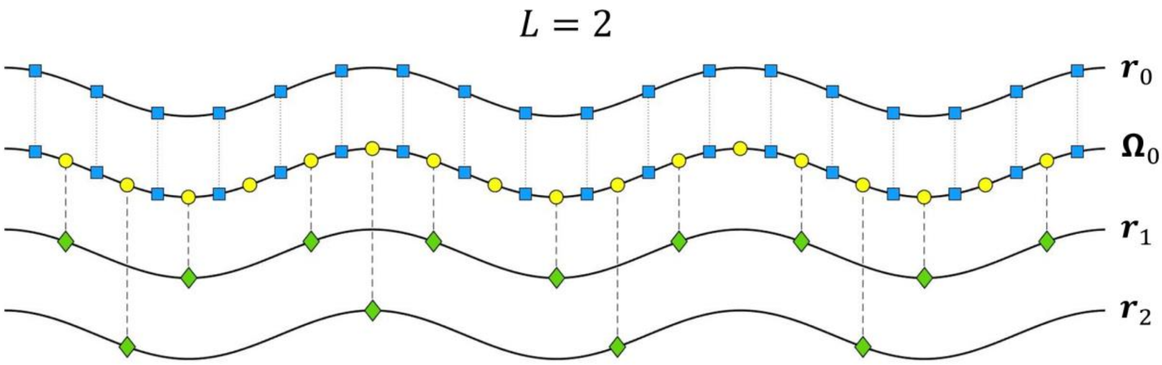


Figure 3.8 – Flowchart of the Fast Factorized Back Projection (FFBP) algorithm: (a) an overview, highlighting the parallel processing; (b) inside the FFBP block, showing the recursive steps with a parent-child dynamic. Source: Góes *et al.*, 2021 [14].

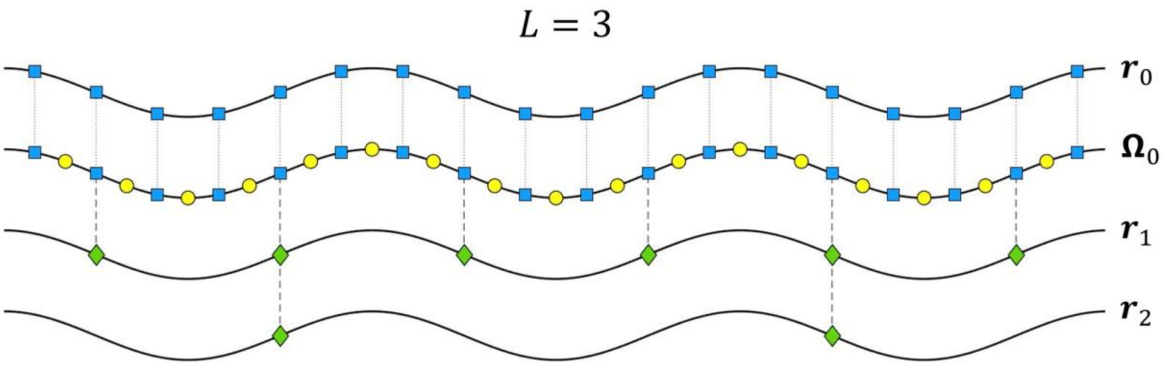
3.2.1 Defining Child Subapertures

The proposed FFBP algorithm employs an original method for defining child subapertures. This method is based on data mapping and does not depend on the flight path. Instead, it depends on a few observations: the flight path follows a continuous line, no matter how complex its geometry; the radar positions over the flight path are usually sampled at regular intervals; the distance between consecutive radar positions is nearly regular if the radar speed is constant.

Figure 3.9 describes the proposed method for defining child subapertures. Blue squares represent the actual radar root positions ($\boldsymbol{r}_0$), yellow circles represent the midpoints between them ($\hat{\boldsymbol{r}}_0$), and green diamonds represent the child subaperture centers at the n^{th} node ($\boldsymbol{r}_n$). This method has two particular cases and one general case, depending on the number of subapertures combined at each recursion (L).



(a)



(b)

Figure 3.9 – Defining child subapertures for (a) $L = 2$ and (b) $L = 3$. The blue squares, yellow circles, and green diamonds represent the radar root positions, the midpoints between them, and the child subaperture centers, respectively. Source: Góes *et al.* [13] © 2020 IEEE.

3.2.1.1 Case 1: L is an even number

When L is even, $\boldsymbol{r}_n$ is always a subset of $\hat{\boldsymbol{r}}_{\mathbf{0}}$ (Figure 3.9(a)). Hence:

$$\boldsymbol{r}_n(k) = \hat{\boldsymbol{r}}_0\left(\frac{(2k+1)L^n}{2} - 1\right) \tag{3.8}$$

where $k = 0, 1, \dots, K_n - 1$, with $K_n = K_0/L^n$ being the number of child subapertures at the n^{th} node, and with K_0 being the number of radar root positions. In this way, the calculation of mean positions is only performed in the root node when deviation errors are the least significant.

3.2.1.2 Case 2: L is an odd number

When L is odd, $\boldsymbol{r}_n$ is always a subset of $\boldsymbol{r}_0$ (Figure 3.9(b)). In addition, if $\boldsymbol{r}_{n-1}$ is the set of parent subaperture centers from the previous node, then $\boldsymbol{r}_n$ is also a subset of $\boldsymbol{r}_{n-1}$. Thus, either of the following expressions can be used:

$$\boldsymbol{r}_n(k) = \boldsymbol{r}_0\left(\frac{(2k+1)L^n - 1}{2}\right) \tag{3.9}$$

$$\boldsymbol{r}_n(k) = \boldsymbol{r}_{n-1}\left(\frac{(2k+1)L - 1}{2}\right) \tag{3.10}$$

where $k = 0, 1, \dots, K_n - 1$.

3.2.1.3 General case

In Figure 3.9, $\boldsymbol{\Omega}_0 = \boldsymbol{r}_0 \cup \hat{\boldsymbol{r}}_0$. The set $\boldsymbol{\Omega}_0$ can be also defined as:

$$\boldsymbol{\Omega}_0(i) = \boldsymbol{r}_0\left({}^{i}\!/_{2}\right) \tag{3.11}$$

where $i = 0, 1, \dots, 2(K_0 - 1)$. Therefore, for any value of L, $\boldsymbol{r}_n$ is always a subset of $\boldsymbol{\Omega}_0$:

$$\boldsymbol{r}_n(k) = \boldsymbol{\Omega}_0\big((2k+1)L^n - 1\big) \tag{3.12}$$

where $k = 0, 1, \dots, K_n - 1$. Note that for all k, if L is odd, then the argument on the right is always even, and vice versa.

3.2.2 Generating Child Subimages

Generating child subimages requires a flexible tree structure. The first reason is that the FFBP algorithm must provide both 2D and 3D images. Though this work focuses on SAR tomography, there are many applications for 2D SAR. The second reason is that the FFBP algorithm must deal with non-uniform resolutions. For instance, Spiral SAR has better horizontal (x, y) than vertical (z) resolution.

The FFBP algorithm uses an original tree structure called the modified Morton curve. It is a space-filling tree structure that arranges multi-dimensional data into a 1D curve that follows a Z pattern, much like the original Morton order curve [40], [41]. The modification, however, makes it more flexible, allowing for different splitting schemes beyond splitting in two in each direction at every recursion. For example, Figure 3.10 shows the first and the second recursions of the modified Morton order curve with a (3 × 3 × 2) split.

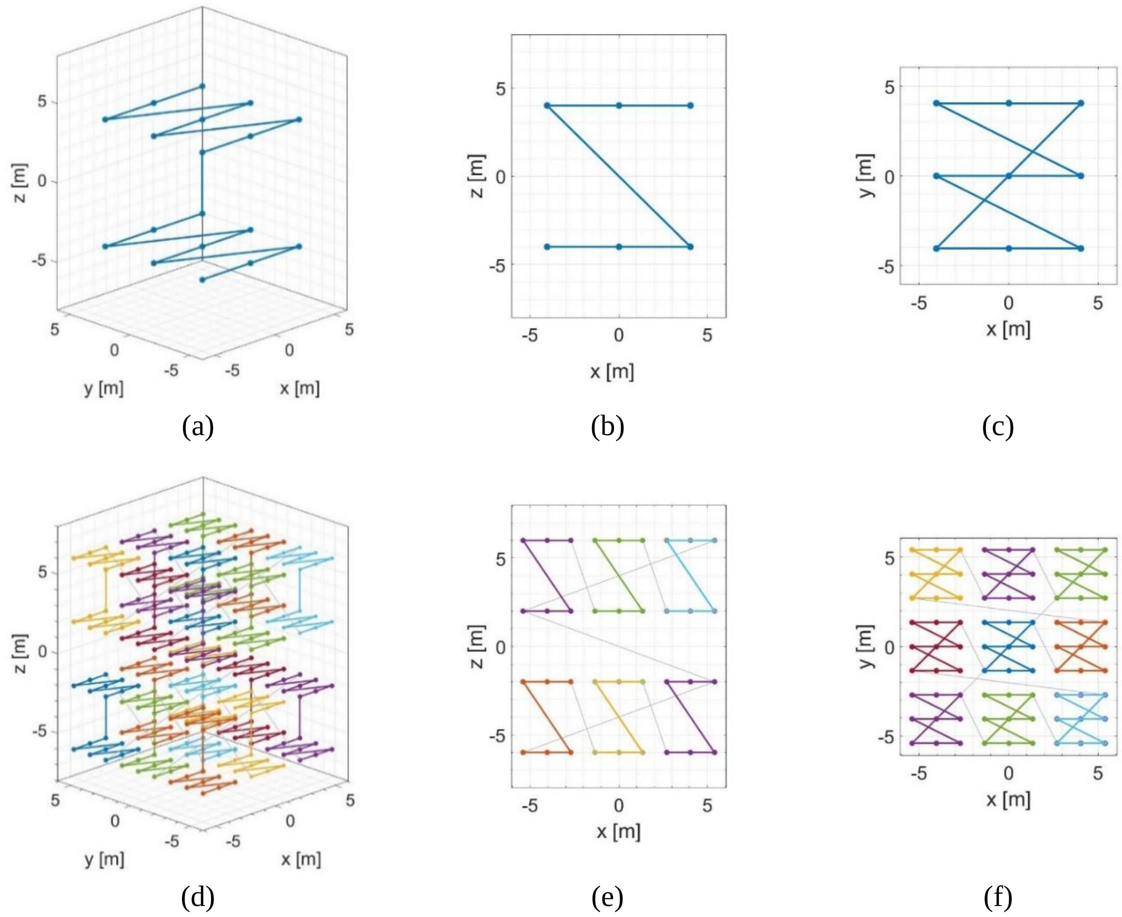


Figure 3.10 – The modified Morton order curve with a (3 × 3 × 2) split: (a,d) perspective, (b,e) front, and (c,f) top views for the (a,b,c) first and (d,e,f) second recursions. Source: Góes *et al.* [13] © 2020 IEEE.

The splitting scheme is created by a function executed in the preparation step (Figure 3.8(a)), referred to as the splitting function. The splitting scheme is a table with as many lines as the number of recursion steps and whose columns contain the number of divisions in the x, y, and z directions (D_x, D_y, and D_z). These quantities are obtained from the dimension and resolution of the output image, the first split into image blocks, and the number of subapertures combined at each recursion L. When working with 2D data, i.e., images with zero thickness, the splitting function makes $D_z = 1$ for all recursions.

After retrieving the splitting scheme for the current recursion, the algorithm finds all possible values of x, y, and z coordinates for the child subimage centers in a local coordinate system with the origin at the parent subimage center. Now, let us define A_{x_n}, A_{y_n} and A_{z_n} as the dimensions of the child subimage at the n^{th} node, then:

$$x(d_x) = A_{x_n}\left(d_x - \frac{(D_x - 1)}{2}\right), \quad A_{x_n} = \frac{A_{x_{n-1}}}{D_x} \tag{3.13}$$

$$y(d_y) = A_{y_n}\left(d_y - \frac{(D_y - 1)}{2}\right), \quad A_{y_n} = \frac{A_{y_{n-1}}}{D_y} \tag{3.14}$$

$$z(d_z) = A_{z_n}\left(d_z - \frac{(D_z - 1)}{2}\right), \quad A_{z_n} = \frac{A_{z_{n-1}}}{D_z} \tag{3.15}$$

where $d_x = 0, 1, \ldots, D_x - 1$, $d_y = 0, 1, \ldots, D_y - 1$ and $d_z = 0, 1, \ldots, D_z - 1$.

Next, the possible values of x, y, and z are arranged in a pattern similar to a truth table in digital systems theory to construct a Z-shaped curve of coordinates $\tilde{x}$, $\tilde{y}$, and $\tilde{z}$ (see

Table 3.1). Then, the position of each child subimage center $\boldsymbol{h}_n(c)$ is given by:

$$\boldsymbol{h}_n(c) = [\tilde{x}(d) \quad \tilde{y}(d) \quad \tilde{z}(d)] + \boldsymbol{h}_{n-1}(p) \tag{3.16}$$

$$c = pD_n + d \tag{3.17}$$

where $\boldsymbol{h}_{n-1}(p)$ is the parent subimage center, $d = 0, 1, \ldots, D_n - 1$, $D_n = D_x D_y D_z$ is the number of children generated by each parent, c refers to a child subimage, and p refers to a parent subimage.

The positions $\boldsymbol{h}_{n-1}(p)$ and $\boldsymbol{h}_n(c)$ do not contain any topographic information. Thus, the terrain height H_{DEM} needs to be interpolated from a DEM. Finally, the actual position of the child subimage $\tilde{\boldsymbol{h}}_{n,c}$ is:

$$\tilde{\boldsymbol{h}}_{n,c} = \boldsymbol{h}_n(c) + [0 \quad 0 \quad H_{DEM}(\boldsymbol{h}_n(c))] \tag{3.18}$$

Table 3.1 – Order of arrangement of the x, y, and z coordinates of the child subimage centers in a modified Morton order curve with a (3 × 3 × 2) split. Source: Góes *et al.*, 2021 [14].

$\boldsymbol{d}$	$\tilde{\boldsymbol{x}}(\boldsymbol{d})$	$\tilde{\boldsymbol{y}}(\boldsymbol{d})$	$\tilde{\boldsymbol{z}}(\boldsymbol{d})$	$\boldsymbol{d}$	$\tilde{\boldsymbol{x}}(\boldsymbol{d})$	$\tilde{\boldsymbol{y}}(\boldsymbol{d})$	$\tilde{\boldsymbol{z}}(\boldsymbol{d})$
0	$x(0)$	$y(0)$	$z(0)$	9	$x(0)$	$y(0)$	$z(1)$
1	$x(1)$	$y(0)$	$z(0)$	10	$x(1)$	$y(0)$	$z(1)$
2	$x(2)$	$y(0)$	$z(0)$	11	$x(2)$	$y(0)$	$z(1)$
3	$x(0)$	$y(1)$	$z(0)$	12	$x(0)$	$y(1)$	$z(1)$
4	$x(1)$	$y(1)$	$z(0)$	13	$x(1)$	$y(1)$	$z(1)$
5	$x(2)$	$y(1)$	$z(0)$	14	$x(2)$	$y(1)$	$z(1)$
6	$x(0)$	$y(2)$	$z(0)$	15	$x(0)$	$y(2)$	$z(1)$
7	$x(1)$	$y(2)$	$z(0)$	16	$x(1)$	$y(2)$	$z(1)$
8	$x(2)$	$y(2)$	$z(0)$	17	$x(2)$	$y(2)$	$z(1)$

3.2.3 Computing Child SAR Data

The child SAR data are both an output of the current recursion and an input for the next. For this reason, multiple range samples are required until the second to last recursion. Also, computing each child SAR datum requires an additional slant range distance, calculated from the range sample to the corresponding child subaperture. Nevertheless, of all steps in Figure 3.8(b), computing child SAR data is the one that most closely resembles the BP algorithm. Its process is illustrated in Figure 3.11.

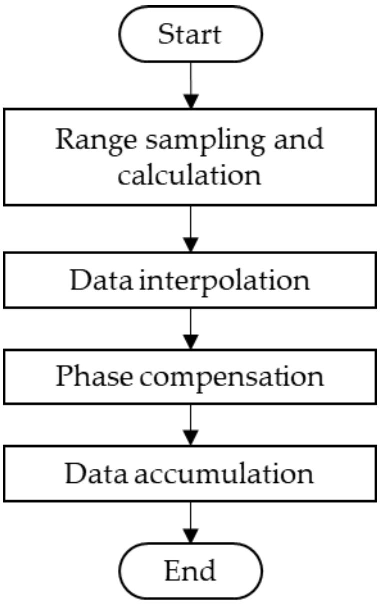


Figure 3.11 – Flowchart of the process for computing child SAR data. Source: Góes *et al.*, 2021 [14].

3.2.3.1 Range sampling and calculation

Range samples are collected along a line defined by the center of the child subaperture $\boldsymbol{r}_n(k)$ and the center of the child subimage $\widetilde{\boldsymbol{h}}_{n,c}$. A sample is always taken at $\widetilde{\boldsymbol{h}}_{n,c}$. Except for the last recursion, other samples are taken along the sphere's diameter that circumscribes the child subimage, as depicted in Figure 3.12. Also, the range sampling interval is the same for all recursions. It is calculated in the preparation step (Figure 3.8(a)) and is equal to the resulting range bin spacing after upsampling the root SAR data.

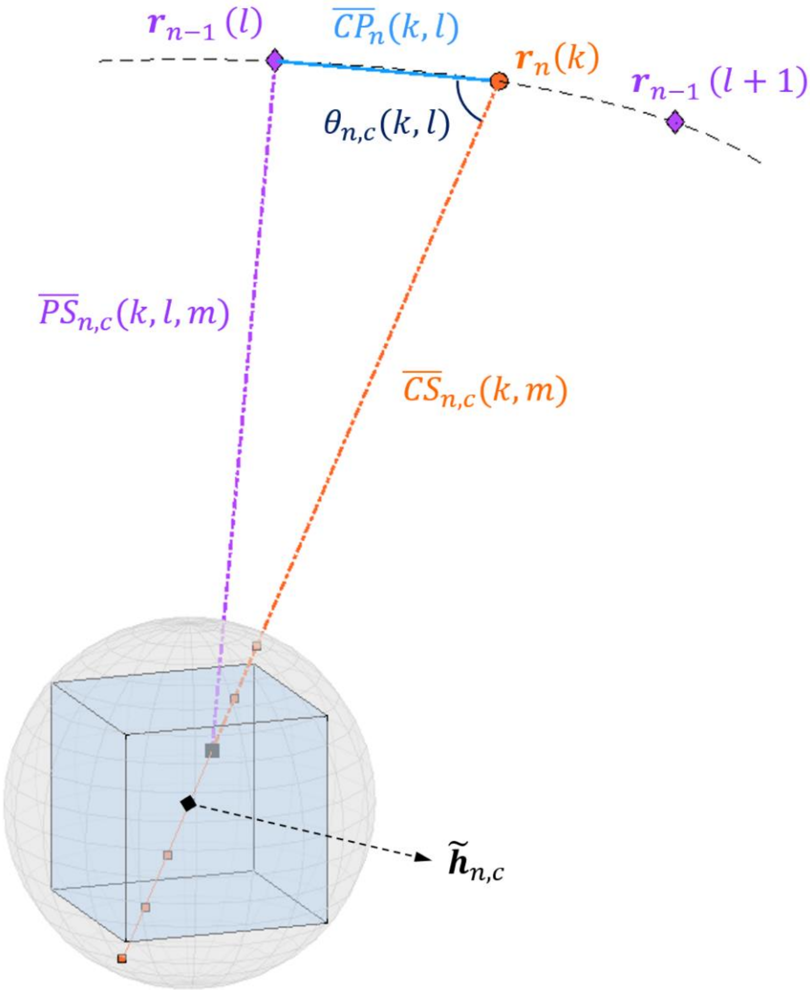


Figure 3.12 – The range samples at a child subimage and the geometry for calculating distances between a range sample, a child subaperture, and a parent subaperture. Source: Góes *et al.*, 2021 [14].

Figure 3.12 also shows a triangle composed of the following vertices [30]:

- C: the child subaperture center $\boldsymbol{r}_n(k)$;
- P: the parent subaperture center $\boldsymbol{r}_{n-1}(l)$;
- S: the m^{th} data sample within a child subimage centered at $\widetilde{\boldsymbol{h}}_{n,c}$.

The side $\overline{CP}_n(k,l)$ is determined by analytic geometry:

$$\overline{CP}_n(k,l) = \|\boldsymbol{r}_{n-1}(l) - \boldsymbol{r}_n(k)\| \tag{3.19}$$

where $l \in \Lambda_{n,k} = \{kL + b | b = 0, 1, \dots, L-1\}$ is the set of parent subapertures associated with the k^{th} child subaperture.

Then, the side $\overline{CS}_{n,c}(k,m)$ is also determined by analytic geometry. Let M_n be the number of range samples at the n^{th} node, thus:

$$\overline{CS}_{n,c}(k,m) = \left\|\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)\right\| + \delta_{sr}\left(m - \frac{(M_n - 1)}{2}\right) \tag{3.20}$$

where $m = 0, 1, \dots, M_n - 1$, δ_{sr} is the range sampling interval.

Next, the side $\overline{PS}_{n,c}(k,l,m)$ is calculated by the law of cosines:

$$\begin{aligned}&\overline{PS}_{n,c}(k,l,m)\\ &= \sqrt{\overline{CP}_n(k,l)^2 + \overline{CS}_{n,c}(k,m)^2 - 2\overline{CP}_n(k,l)\overline{CS}_{n,c}(k,m)\cos\big(\theta_{n,c}(k,l)\big)}\end{aligned} \tag{3.21}$$

$$\cos\big(\theta_{n,c}(k,l)\big) = \frac{\boldsymbol{r}_{n-1}(l) - \boldsymbol{r}_n(k)}{\overline{CP}_n(k,l)} \cdot \frac{\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)}{\left\|\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)\right\|} \tag{3.22}$$

3.2.3.2 Data interpolation

From the distance $\overline{PS}_{n,c}(k,l,m)$, the FFBP algorithm can retrieve the parent data $s_{n-1}\big(l, \nu_{n,c}(k,l,m), p\big)$, associated with the m^{th} data sample. The fractional index $\nu_{n,c}(k,m,l)$ is given by [30]:

$$\nu_{n,c}(k,l,m) = \frac{\overline{PS}_{n,c}(k,l,m) - \overline{CS}_{n-1,p}(l,0)}{\delta_{sr}} \tag{3.23}$$

where $\overline{CS}_{n-1,p}(l,0)$ is the slant range from the parent subaperture to the first sample in the parent data. Then, the value $s_{n-1}\big(l, \nu_{n,c}(k,l,m), p\big)$ is determined via linear interpolation.

3.2.3.3 Phase compensation

To ensure good image quality, the FFBP uses the phase compensation term [39]:

$$\varphi_{n,c}(k,l,m) = \frac{4\pi}{\lambda}\left[\overline{PS}_{n,c}(k,l,m) - \overline{CS}_{n,c}(k,m)\right] \tag{3.24}$$

where λ is the radar wavelength.

3.2.3.4 Data accumulation

Finally, the child datum $s_n(k,m,c)$ is given by the coherent sum [30]:

$$s_n(k,m,c) = \sum_{l\in\Lambda_{n,k}} s_{n-1}\big(l, \nu_{n,c}(k,l,m), p\big)\, e^{j\varphi_{n,c}(k,l,m)} \tag{3.25}$$

Each of the indices k, m, and l corresponds to a different matrix dimension. Note that none of the variables denoting position (indicated in bold letters) depend on the data sample index m, so there is no need for a fourth matrix dimension to account for the (x,y,z) triplets.

3.2.4 Coherent Sum of Remaining Data

After reaching the final recursion step ($n = N$), the remaining subapertures are coherently combined. This process has two main differences compared to computing the child SAR data: there is only one set of subapertures to take into account, $\boldsymbol{r}_N(k)$; there is only one sample at each child subimage ($M_N = 1$). Therefore, the resulting SAR data are:

$$s(c) = \sum_{k} s_N(k,0,c)e^{j\tilde{\varphi}_c(k)} \tag{3.26}$$

$$\tilde{\varphi}_c(k) = \frac{4\pi}{\lambda}\left\|\tilde{\boldsymbol{h}}_{N,c} - \boldsymbol{r}_N(k)\right\| \tag{3.27}$$

3.2.5 Data Mapping from 1D to 2D/3D

Finally, the resulting serial data $s(c)$ are mapped into a data matrix. To retrieve the 3D matrix subscripts (u,v,w) from the index c of the modified Morton order curve, the FFBP algorithm uses recurrent sequences. These sequences are also built in a parent-child dynamic to allow flexible splitting schemes. Let q_{x_n}, q_{y_n}, and q_{z_n} be the recurrent sequences of the n^{th} recursion, then:

$$q_{x_0}(0) = q_{y_0}(0) = q_{z_0}(0) = 0 \tag{3.28}$$

$$q_{x_n}(uD_x + d_x) = D_n q_{x_{n-1}}(u) + d_x \tag{3.29}$$

$$q_{y_n}(vD_y + d_y) = D_n q_{y_{n-1}}(v) + d_y D_x \tag{3.30}$$

$$q_{z_n}(wD_z + d_z) = D_n q_{z_{n-1}}(w) + d_z D_x D_y \tag{3.31}$$

Recall that $d_x = 0, 1, \ldots, D_x - 1$, with D_x being the number of divisions in the x direction, and the same for d_y, d_z, D_y and D_z.

Then, the mapping $c \to (u, v, w)$ follows the relationship:

$$c = q_{x_N}(u) + q_{y_N}(v) + q_{z_N}(w) \tag{3.32}$$

Expression (3.32) maps the modified Morton order curve into a 2D/3D matrix.

Figure 3.13 demonstrates how equations (3.28-3.32) correspond to the curve shown in Figure 3.10(d,e,f). The sequences q_x and q_y are indicated on the axes, and each panel corresponds to a different element of q_z. The child subimage index starts with $c = 0$ at the bottom left corner of Figure 3.13(a), then moves back and forth between the layers $q_z = 0$ and $q_z = 9$ until reaching $c = 161$ at the top right corner of Figure 3.13(b). Then, it continues at the bottom left corner of Figure 3.13(c), going back and forth between $q_z = 162$ and $q_z = 171$ up until the end, at the top right corner of Figure 3.13(d).

3.2.6 FFBP Algorithm Improvements

There were different versions of the FFBP algorithm throughout this work. Section 3.3 presents simulation results obtained with a preliminary version of the FFBP algorithm. Although fully functional, it did not take advantage of the inherently parallel nature of FFBP algorithms. Also, it did not employ vectorized variables, making it inefficient when operating with actual SAR data.

Therefore, vectorized variables were introduced in the second version of the FFBP algorithm to improve the processing time. At first, there were six concatenated for-loops, one for each of the indices in Subsections 3.2.1-3.2.5 (n, p, c, k, l and m). Afterward, there were only two (n and c). This change made the processing time drop more than 20 times, from 11 h 30.5 min to 4.5 min. Both cases used the same data and setup: a 2D image, processed with $L = 5$ and a first split of (12 × 6 × 1).

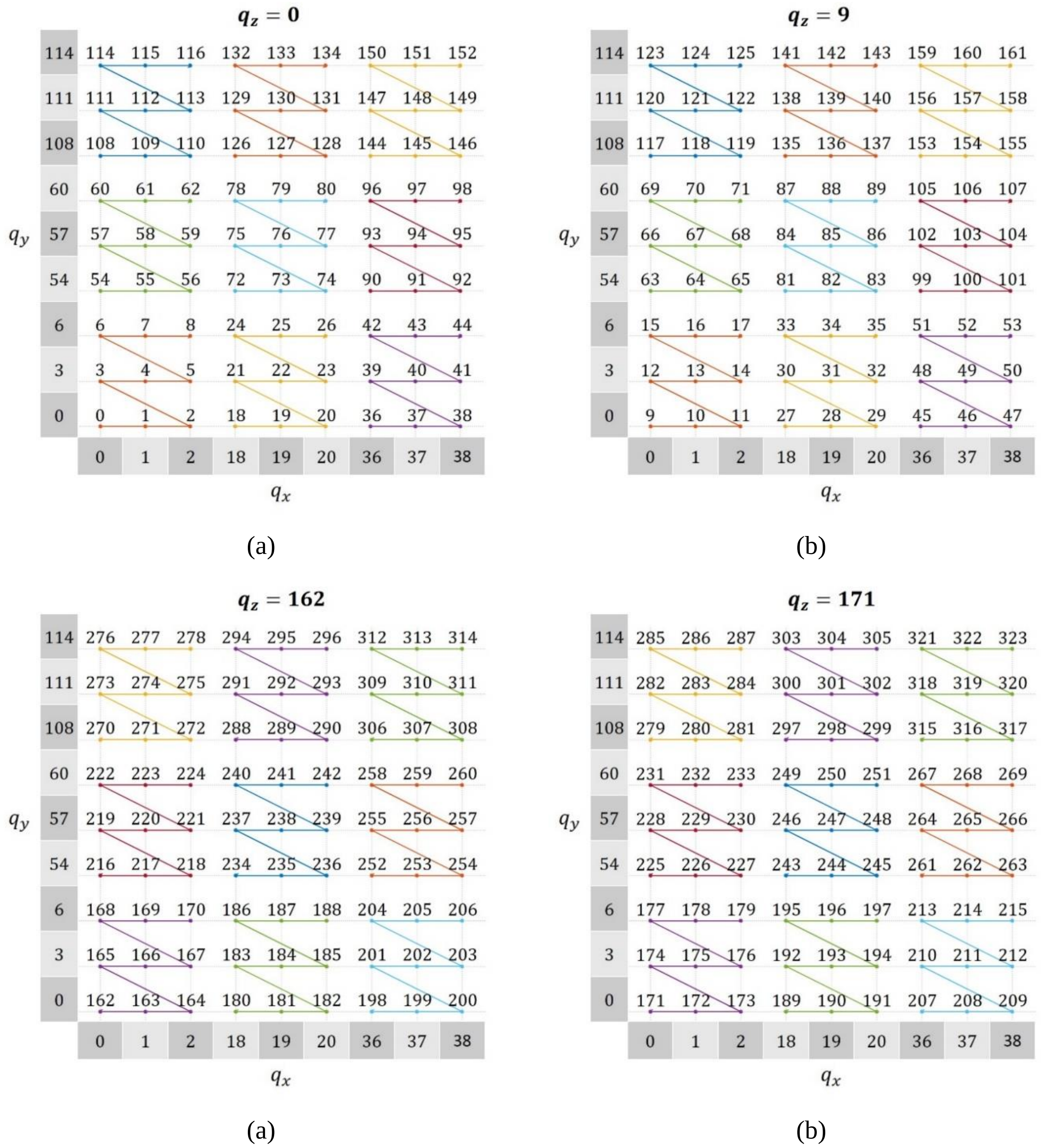


Figure 3.13 – Child subimage indices c at their positions over the modified Morton order curve with (a) $q_z = 0$, (b) $q_z = 9$, (c) $q_z = 162$, and (d) $q_z = 171$. The recurrent sequences q_x and q_y are displayed on the axes. Source: Góes *et al.*, 2021 [14].

However, when processing 3D images, that version still proved inefficient in memory consumption. So, the division into image blocks and parallel computing were introduced to mitigate this issue. Moreover, the splitting function was introduced in this more consolidated version of the FFBP algorithm. Prior to that, the splitting scheme was done manually. Section 3.4 presents experimental results obtained with the latest version of the FFBP algorithm, which is openly available at [15].

3.3 PROOF OF CONCEPT

This section presents some simulation results carried out as proof of concept. The aim is to show that the FFBP algorithm can process images in full-3D for nonlinear flight paths. All results were obtained with the preliminary version of the FFBP algorithm, which employed neither parallel computing functions nor vectorized variables. For comparison purposes, the BP algorithm was written with these same conditions. All algorithms presented in this section were written in MATLAB R2018a, and all simulations were executed on an Intel(R) Core(TM) i7-7700 CPU (3.60 GHz) with 64 GB RAM.

3.3.1 Point Spread Function

The first simulated scenario consisted of a drone-borne SAR system operating at the P-band and performing a spiral flight path with constant radius and speed (see Figure 3.14). Table 3.2 shows the simulated radar acquisition parameters. There were nine isotropic point targets in the imaged volume: one at the origin of a cartesian coordinate system and the others at the vertices of a cube of 8 m, centered at the origin.

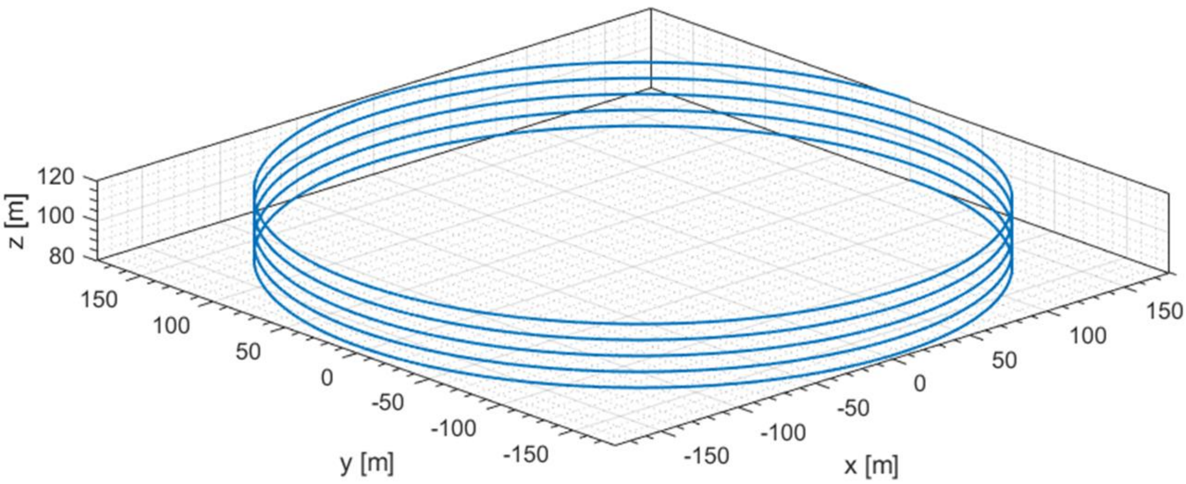


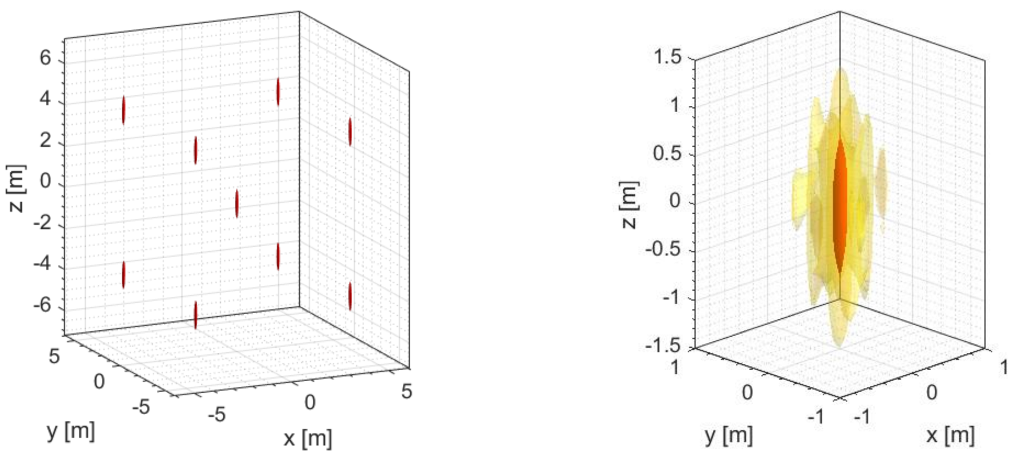
Figure 3.14 – Simulated spiral flight path. Source: Góes *et al.* [13] © 2020 IEEE.

The FFBP algorithm was set up with six recursions. The subapertures were combined in groups of $L = 3$, and the range sampling was 0.125 m. There were 174,960 ($3^6 \times 240$) radar root positions, and the imaged volume was $12.15 \times 12.15 \times 14.4$ m^3. The first recursion did not split the root subimage, i.e., a $(1 \times 1 \times 1)$ split; the following four recursions performed $(3 \times 3 \times 2)$ splits (see Figure 3.10); the last recursion split each subimage into $(3 \times 3 \times 3)$ smaller ones. Finally, the remaining 240 data blocks were coherently combined.

Table 3.2 – Simulated radar acquisition parameters. Source: Góes *et al.* [13] © 2020 IEEE.

Parameters		**Values**	**Units**
Radar	Wavelength	0.75	m
	Bandwidth	150	MHz
	Range resolution	1	m
	Pulse repetition frequency	200	Hz
Flight path	Radius	180	m
	Height at the top	120	m
	Height at the base	80	m
	Number of turns	5	-
	Drone speed	6.5	m/s

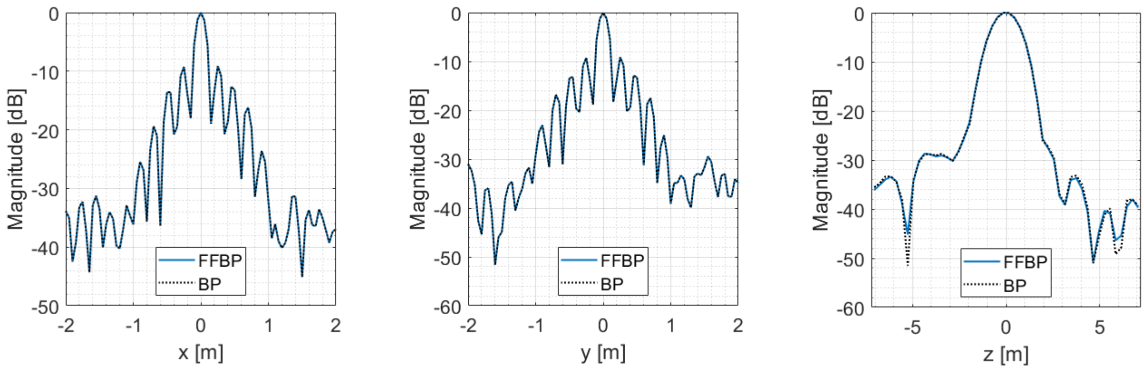
The 3D output image had $243 \times 243 \times 48$ voxels of dimension $5 \times 5 \times 30$ cm^3. Figure 3.15(a) shows the distribution of all nine targets after processing with the FFBP algorithm. Figure 3.15(b) shows a closer caption of the central target. It depicts a -3 dB isosurface in opaque red and -13 dB isosurfaces in translucent yellow.



(a) (b)

Figure 3.15 – Output image processed with the FFBP algorithm: (a) -3 dB isosurfaces for the entire imaged volume; (b) a -3 dB isosurface in opaque red, and -13 dB isosurfaces in translucent yellow for the target at the origin. Source: Góes *et al.* [13] © 2020 IEEE.

Figure 3.16 displays the point spread function for the FFBP and the BP algorithms. Note that the plot curves are nearly the same for both algorithms. The (x, y) plane had a refined resolution of 16 cm but a poor PSLR of 9.1 dB. In contrast, the z direction presented a coarser resolution of 1.53 m with a better PSLR of 28.7 dB. Furthermore, the mean phase error was ~10^{-4} rad, and the standard deviation was 0.12 rad (7°), more than three times lower than the recommended threshold of π/8 rad (22.5°) [31]. The mean and the standard deviation of the magnitude error were, respectively, 0.1 dB and 0.9 dB. Finally, the degree of coherence between the FFBP and the BP images was 0.9993. Table 3.3 summarizes these results.



(a) (b) (c)

Figure 3.16 – Comparison between the FFBP and the BP algorithms. Point spread function: normalized magnitude in dB against (a) x, (b) y, and (c) z. Source: Góes *et al.* [13] © 2020 IEEE.

Table 3.3 – Summary of results for the point spread function. Source: Góes *et al.* [13] © 2020 IEEE.

Parameters		Values	Units
Image quality	3 dB resolution in the (x,y) plane	16	cm
	3 dB resolution in the z-direction	1.53	m
	PSLR in the (x,y) plane	9.1	dB
	PSLR in the z-direction	28.7	dB
Image comparison	Mean phase error	~10^{-4}	rad
	Standard deviation of the phase error	0.12	rad
	Mean magnitude error	0.1	dB
	Standard deviation of the magnitude error	0.9	dB
	Degree of coherence	0.9993	-
Processing time	FFBP	**3.45**	h
	BP	**39.09**	h

The FFBP algorithm took 3.45 hours (~0.14 days) to process, while the BP algorithm took 39.09 hours (~1.6 days). This difference represents a 91.1 % reduction in processing time (or an 11.3 speed-up factor). This result concerns the FFBP using the general case for defining child subapertures (3.12), which was 5.5 minutes faster than using the particular case (3.10), and 19.1 minutes faster than calculating weighted mean positions [30]:

$$\boldsymbol{r}_n(k) = \frac{1}{L} \sum_{l \in \Lambda_{n,k}} \boldsymbol{r}_{n-1}(l) \tag{3.33}$$

where $\Lambda_{n,k}$ is the set of parent subapertures that combine to create the k^{th} child subaperture.

The low phase error was due, in part, to the choice of the subimages dimensions, as the analysis in [31] suggests (see Subsection 3.1.3). However, without (3.24), the results would be far worse. The standard deviation of the phase error would be 1.63 rad (93°), yielding a degree of coherence of 0.29. Indeed, Figure 3.17 shows the plane sections at $z = 0$ m and $z = 4$ m for the BP and the FFBP output images, processed with and without phase compensation. Note that, without phase compensation, only the target at the center is processed correctly. The errors increase as we move away from the origin, and the targets at $z = 4$ m are not well focused.

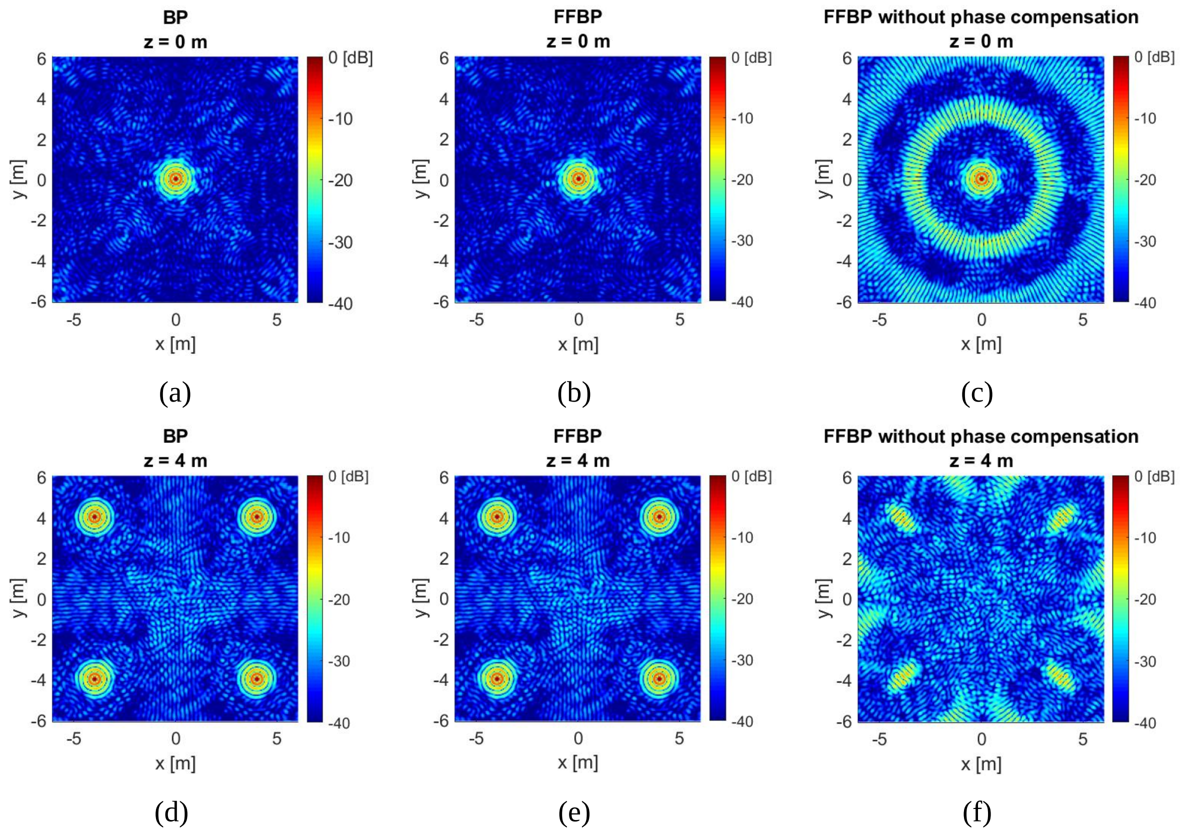


Figure 3.17 – Plane sections at (a,b,c) $z = 0$ m and (d,e,f) $z = 4$ m for the output images processed with (a,d) the BP, (b,e) the FFBP and (c,f) the FFBP without phase compensation. Normalized magnitudes in dB.

3.3.2 Random Phase Error

The second simulated scenario also used the spiral flight path depicted in Figure 3.14 with the radar acquisition parameters of Table 3.2. However, a random phase error was added to the simulated radar data to represent motion data inaccuracies. The total relative position error of the drone-borne SAR system was measured in [12] and had a standard deviation of 7.4 mm, corresponding to a phase error standard deviation of 0.12 rad (7°) at the 0.75 m wavelength. Thus, the simulated radar data was perturbed by a random phase error. This error had a normal distribution with zero mean and the standard deviation above.

Real radar cross-section data can be modeled as a cloud of isotropic point targets [42]. For that reason, a random distribution of point targets was adopted for this scenario. The initial target grid had 81 × 81 × 9 points with 0.15 m × 0.15 m × 1.5 m spacing. Each position corresponded to a Bernoulli random variable with a 0.001 probability of being a point target. The resulting distribution had 85 targets in total. It is illustrated in Figure 3.18(a,b,c). Figure 3.18(d,e,f) shows the FFBP output image, depicting -6dB isosurfaces.

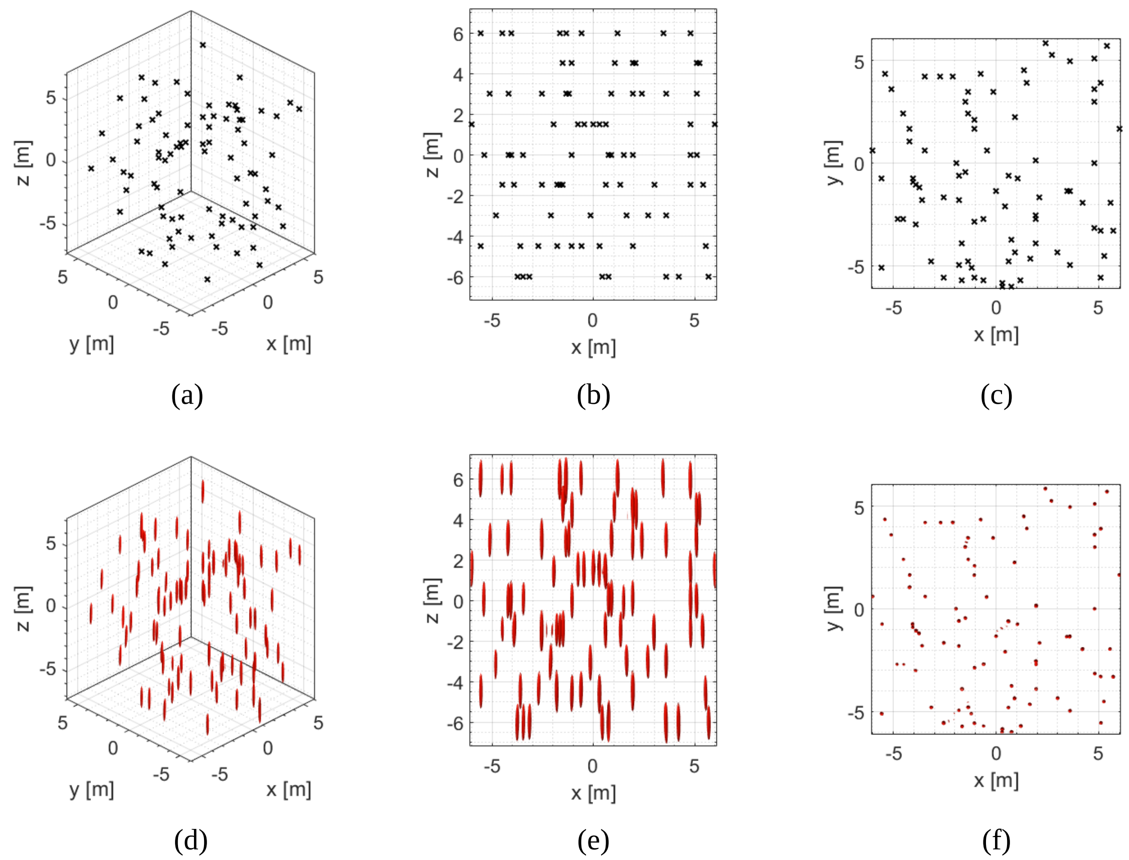
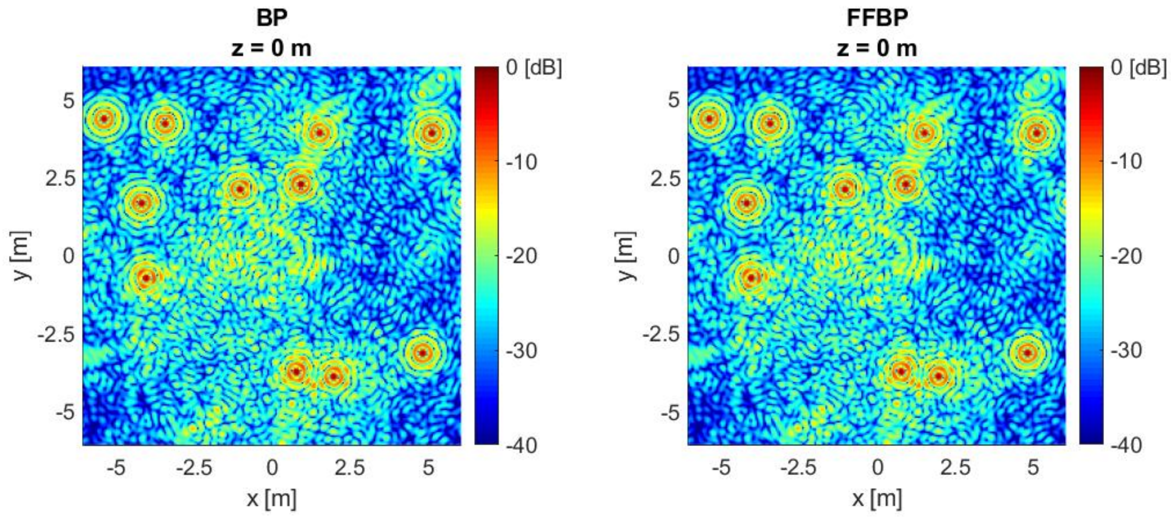


Figure 3.18 – Comparison between input and output target distributions: (a,d) perspective, (b,e) front and (c,f) top views for (a,b,c) the input point targets and (d,e,f) -6 dB isosurfaces of the FFBP output image. Source: Góes *et al.* [13] © 2020 IEEE.

Figure 3.19 shows a plane section of the FFBP and the BP output images. The two figures were nearly identical, thus suggesting that both algorithms were equally affected by the random phase error added to the simulated radar data. The degree of coherence between the two output images was 0.9992. The phase error had a ~10^{-4} rad mean and a 0.10 rad (6°) standard deviation. The mean magnitude error was 0.1 dB, and the standard deviation was 0.8 dB. The processing time was 3.66 hours (~0.15 days) for the FFBP and 39.71 hours (~1.7 days) for the BP, corresponding to a 90.8 % reduction of processing time (a 10.8 speed-up factor). A summary is provided in Table 3.4.



(a) (b)

Figure 3.19 – Plane section at $z = 0$ m for (a) the BP and (b) the FFBP output images. Normalized magnitudes in dB. Source: Góes *et al.* [13] © 2020 IEEE.

Table 3.4 – Summary of results with added random phase error. Source: Góes *et al.* [13] © 2020 IEEE.

Parameters		Values	Units
Image comparison	Mean phase error	~10^{-4}	rad
	Standard deviation of the phase error	0.10	rad
	Mean magnitude error	0.1	dB
	Standard deviation of the magnitude error	0.8	dB
	Degree of coherence	0.9992	-
Processing time	FFBP	**3.66**	h
	BP	**39.71**	h

3.3.3 Random Flight Path

The third simulated scenario used a random spiral flight path, whose coordinates were defined by stochastic processes (see Figure 3.20). The distance was fixed from one step to the next, $\Delta r = 0.25$ m, but the direction was random. The change in elevation angle was a uniform random variable, $\Delta\varepsilon \sim U(-\pi/2, \pi/2)$. The change in azimuth angle was a function of the current position in azimuth ($\alpha(i)$) and a uniform random variable centered at $\pi/2$, $\Delta\alpha \sim U(-\pi/8, 9\pi/8)$. Thus:

$$x(0) = 180\,m, \quad y(0) = 0\,m, \quad z(0) = 100\,m \tag{3.34}$$

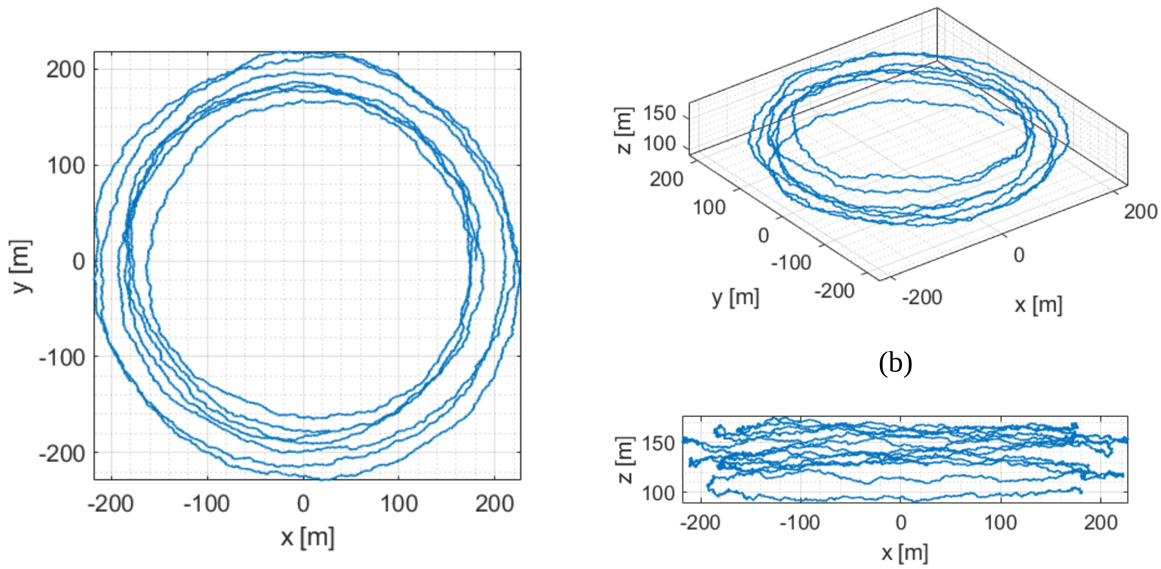
$$x(i+1) = x(i) + \Delta r \cos(\Delta\varepsilon) \cos(\alpha(i) + \Delta\alpha) \tag{3.35}$$

$$y(i+1) = y(i) + \Delta r \cos(\Delta\varepsilon) \sin(\alpha(i) + \Delta\alpha) \tag{3.36}$$

$$z(i+1) = z(i) + \Delta r \sin(\Delta\varepsilon) \tag{3.37}$$

$$\alpha(i) = \tan^{-1}\left(\frac{y(i)}{x(i)}\right) \tag{3.38}$$

The root node had 109,350 ($3^6 \times 150$) radar positions, so only 150 blocks remained at the last step of the FFBP algorithm to be coherently combined. In all other aspects, the setup was the same as in Subsections 3.3.1 and 3.3.2.



(a) (c)

Figure 3.20 – Random spiral flight path: (a) top, (b) perspective, and (c) front views.

This scenario also adopted a random distribution of isotropic point targets. There were 100 targets in total, and their positions were given by a multivariate normal distribution, with zero mean and covariance matrix:

$$\sigma_{xyz} = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 4 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{3.39}$$

Figure 3.21 shows the input distribution of point targets as black dots. Figure 3.21 also shows the output image, processed with the FFBP algorithm, displaying -10 dB isosurfaces in translucent orange.

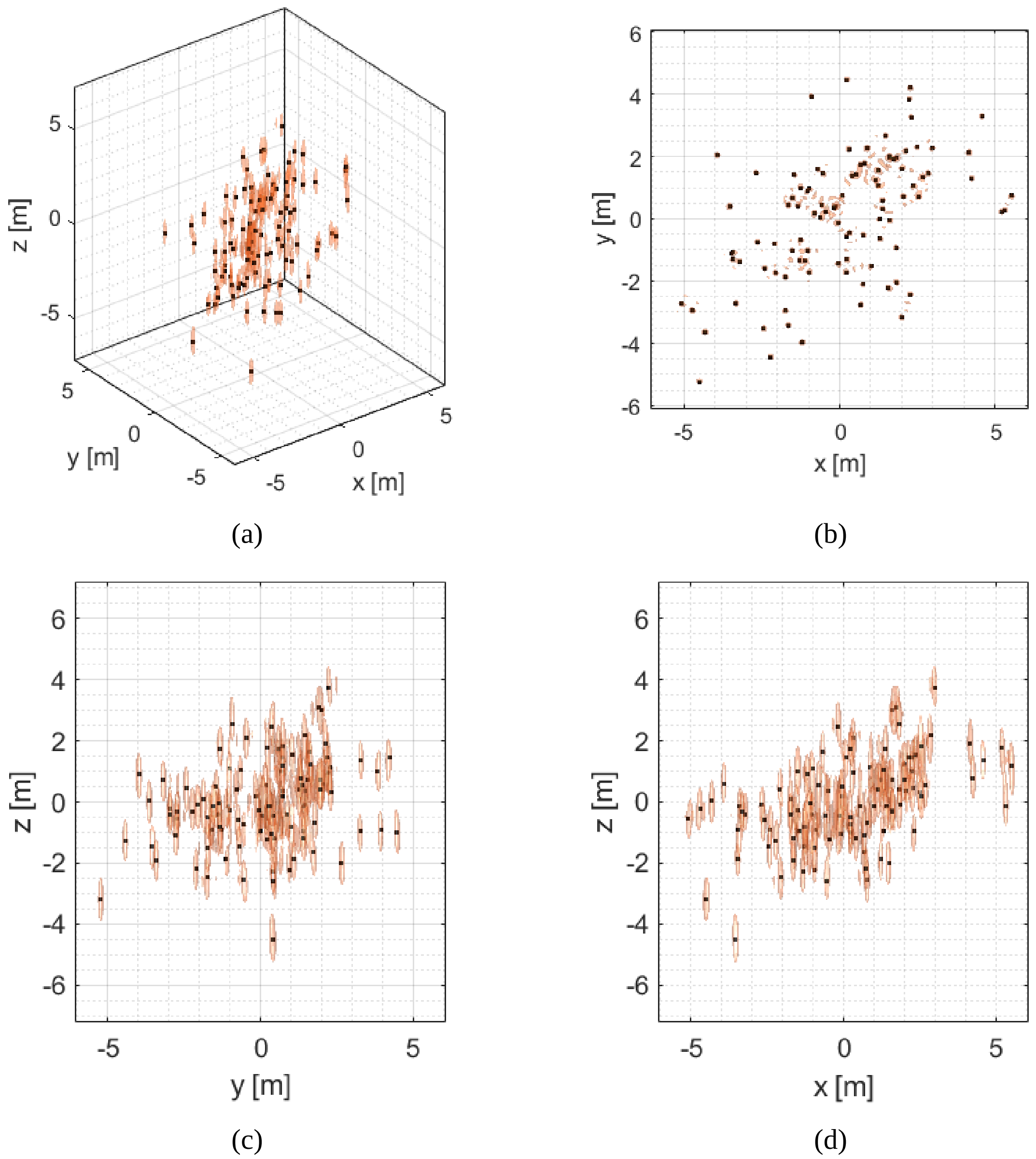
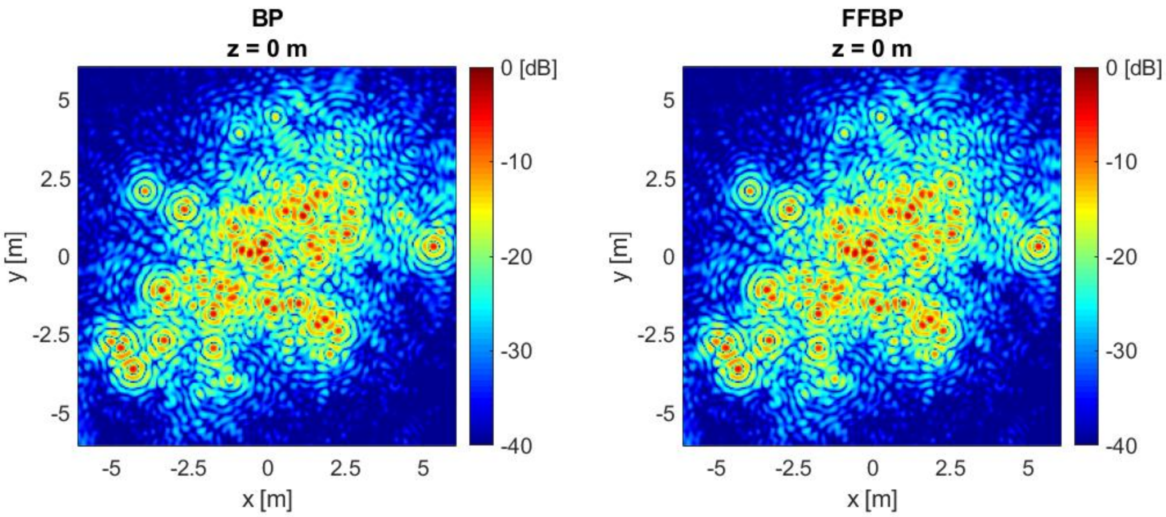


Figure 3.21 – Comparison between target distributions: the black dots are the input point targets; the translucent orange curves are the -10 dB isosurfaces for the FFBP output image. Each panel shows a different view: (a) perspective, (b) top, (c) side, and (d) front.

Figure 3.22 shows a plane section of the FFBP and the BP output images. Again, the two figures were almost identical. The degree of coherence between them was 0.9996. The mean and the standard deviation of the magnitude error were, respectively, 0.03 dB and 0.7 dB. The mean and the standard deviation of the phase error were ~10^{-4} rad and 0.09 rad (5°). A summary is provided in Table 3.5.



(a) (b)

Figure 3.22 – Plane section at $z = 0$ m for (a) the BP and (b) the FFBP output images. Normalized magnitudes in dB.

Table 3.5 also shows that the FFBP algorithm took 2.25 hours (~0.09 days) to process, while the BP algorithm took 26.65 hours (~1.1 days), corresponding to a 90.8 % reduction in processing time (an 11.8 speed-up factor). Note that these processing times are significantly lower than the ones presented in Table 3.3 and Table 3.4. The reason is that the random flight path of Figure 3.20 has 37.5 % fewer radar root positions than the flight path shown in Figure 3.14.

Table 3.5 – Summary of results for the random spiral flight path.

Parameters		Values	Units
Image comparison	Mean phase error	~10^{-4}	rad
	Standard deviation of the phase error	0.09	rad
	Mean magnitude error	0.03	dB
	Standard deviation of the magnitude error	0.7	dB
	Degree of coherence	0.9996	-
Processing time	FFBP	**2.25**	h
	BP	**26.65**	h

3.3.4 Discussion

The FFBP algorithm was tested in three simulation scenarios inspired by the drone-borne SAR system of Section 1.1 and data acquisition techniques that produce high-resolution 3D images [26]. The first scenario had an ideal spiral flight path; the second added a random phase error to the simulated radar data; the third used a random spiral flight path. Furthermore, the imaged volume had clouds of point targets, with different distributions for each scenario. The three scenarios prove that the FFBP algorithm can process data in full-3D for nonlinear flight paths. To the best of the author's knowledge, it is the first algorithm capable of such a feat.

Indeed, for the Multi-Circular SAR presented by Ponce *et al.* [5], different image layers were processed with a 2D FFBP customized for circular flight paths [6]. Ponce *et al*. did not pursue 3D-focusing with their FFBP for practical reasons [5]. Other Multi-Circular SAR solutions used the BP algorithm [7], [8], sparse reconstruction models [5], [9], [10], adaptive imaging [7], [11], or a combination of those. Apart from the choice of algorithm, there are two common approaches: either process each circular flight path independently and merge the outputs [5], [7] or make radial slices of the cylindrical synthetic aperture, process them separately, then combine the results [8]–[11]. For the spiral SAR presented in [12], the whole flight path was processed with the BP algorithm.

Table 3.3 to Table 3.5 encapsulate the simulation results. The FFBP achieved over 90% reduction in processing time without losing image quality. Indeed, the degree of coherence between the FFBP and BP images was nearly equal to one in each scenario. In addition, the phase error's standard deviation was much smaller than the typical $\pi/8$ rad threshold used for assessing FFBP algorithms. The proposed method for defining child subapertures was 8 % faster than the typical calculation of weighted mean positions (3.33).

The speed-up factor could be further increased if another sampling strategy was used, possibly at the expense of a higher phase error. Taking the spheres that circumscribe each subimage into account helped ensure excellent image quality, as the results demonstrated. However, it is not a very efficient approach. In [39], instead of partitioning the range data, a fixed number of pivots is used for combining subapertures. Then, the range data are processed in bulk, using FFT interpolation to increase efficiency.

3.4 PHASE ERROR ANALYSIS

Choosing the best setup when processing SAR data with an FFBP algorithm is not straightforward. In the latest version of the FFBP algorithm, a splitting function creates the splitting scheme. However, the first split into image blocks is still an input provided by the user. So, predicting how this choice will affect the quality of the output image is valuable information.

Thankfully, Ulander *et al*. [31] analyzed the errors introduced by the FFBP algorithm and provided a way to control them (see Subsection 3.1.3). In essence, the range error increases with the length of the child subaperture and the width of the child subimage, and it decreases with the distance between those two entities. Furthermore, the average range error across all subapertures and recursions is proportional to the phase error. Thus, keeping the phase error below a particular threshold requires balancing the increase in subaperture length and the decrease in subimage width.

However, it would not be possible to replicate their analytical development for arbitrary flight paths. Therefore, this thesis takes a different approach. This section uses a hypothesis test to investigate if we can predict the standard deviation of the phase error ($\sigma_{\Delta\varphi}$) from parameters at the start of processing. Specifically, the goal is to test the following hypothesis:

$$\sigma_{\Delta\varphi} \propto \kappa = \frac{4\pi}{\lambda} \cdot \frac{\delta_k \Delta_c}{R_{min}} \tag{3.40}$$

where R_{min} is the shortest distance from the radar to the imaged volume, δ_k is the length of the child subaperture, and Δ_c is the diagonal of the child subimage.

The subimage diagonal replaced the subimage width because the diagonal is more relevant to the FFBP algorithm presented in Section 3.2. Moreover, unlike [31], the FFBP algorithm does not assume a linear flight path to simplify calculations. Thus, expression (3.40) does not consider the effect deviations from a linear flight path would have on the phase error.

3.4.1 The Case Study

For testing the hypothesis (3.40), the case study used SAR data acquired by the drone-borne SAR system of Section 1.1 flying over a eucalyptus plantation with a spiral flight path. Figure 3.23 displays a Google Earth image of the drone flight path over the imaged area; the eucalyptus plantation can be seen on the bottom left. The spacing between the trees was around 3 m. The survey took place in Mogi Guaçu, São Paulo, Brazil, on November 13, 2019. The drone-borne SAR system works with three different frequency bands, but only the results for the P-band were used for the case study. Table 3.6 shows the acquisition parameters.

Figure 3.23 – Google Earth image of the spiral flight path over the imaged area. Source: Góes *et al.*, 2021 [14].

Table 3.6 – Acquisition parameters. Source: Góes *et al.*, 2021 [14].

	Parameters	**Values**	**Units**
Radar	Wavelength	70.54	cm
	Bandwidth	50	MHz
	Slant range resolution	2.4	m
	Pulse repetition frequency	64.95	Hz
Antenna	Antenna horizontal beamwidth	55.9	degrees
	Antenna vertical beamwidth	75.2	degrees
	Antenna depression angle	30	degrees
Flight path	Mean drone speed	8.5	m/s
	Mean flight radius	338	m
	Height at the top	120	m
	Height at the base	79	m
	Number of turns	3	-
	Number of radar root positions	48,128	-

In expression (3.40), the product $\delta_k \Delta_c$ should ideally be constant across all recursions, but in reality, it may vary. So, this product is evaluated in three situations, yielding different values of κ: a value at the first recursion (κ_1), an average value (κ_{avg}), and a maximum value (κ_{max}). The goal is to compare the three situations to make sure that κ_1 is a good choice for predicting $\sigma_{\Delta\varphi}$.

The reason for preferring κ_1 is that it is easier to determine δ_k and Δ_c for the first recursion. For the other recursions, it is the splitting scheme that determines the balance between δ_k and Δ_c. At the start of processing, these values can be set indirectly by:

- Choosing the number of subapertures that are combined at each recursion (L);
- Choosing the first split into image blocks;

Moreover, to vary R_{min}, the analysis selects two image blocks, one close to the edge and one close to the center of the output image (see Figure 3.24).

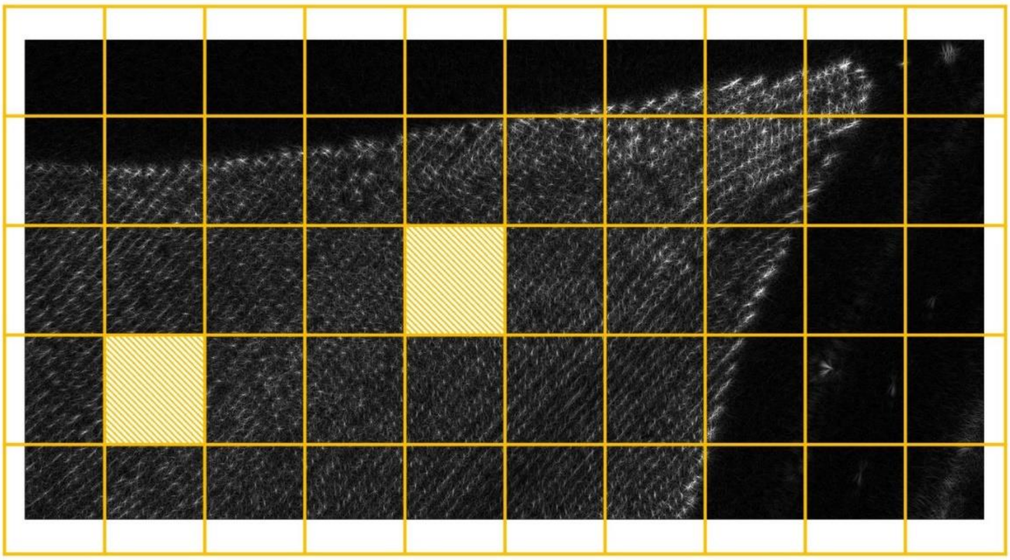


Figure 3.24 – Selected image blocks for analyzing the phase error as a function of processing inputs. Source: Góes *et al.*, 2021 [14].

Table 3.7 shows the set of processing inputs selected for the case study. The splitting scheme in each setup is the same for all image blocks. Thus, the actual number of pixels or voxels might not match the one calculated from the dimension and resolution of the output image. Hence, there are two options: process the image with a different resolution or change the size of the processed image. The second option is adopted because a different resolution would hinder comparing the FFBP and BP images, which is necessary for investigating the hypothesis.

Table 3.7 – Processing inputs for the case study. Source: Góes *et al.*, 2021 [14].

Processing Inputs		**Values**
Dimension of the output image	2D	300×150 m^2
	3D	$300 \times 150 \times 2.4$ m^3
Resolution of the output image	2D	0.2×0.2 m^2
	3D	$0.2 \times 0.2 \times 0.2$ m^3
L		2
		3
		4
		5
First split into image blocks		$8 \times 4 \times 1$
		$12 \times 6 \times 1$
		$16 \times 8 \times 1$
		$20 \times 10 \times 1$
		$24 \times 12 \times 1$

Therefore, the algorithm processes a larger image, and the unnecessary pixels or voxels are disregarded. In a previous version of the FFBP algorithm, the number of image blocks was kept equal to the first split given at the input, but that proved wasteful. Thus, in the current version of the FFBP algorithm, the actual number of image blocks might be larger to minimize waste. Ultimately, this results in a wider variation for Δ_c and R_{min}. The outcome is figuratively represented in Figure 3.24.

Both BP and FFBP algorithms were written in MATLAB R2018a with vectorized variables and parallel computing functions. All data were processed on an Intel(R) Core (TM) i7-7700 CPU (3.60 GHz) with 64 GB RAM.

3.4.2 Phase Error vs. SNR

Pixels close to the noise level could potentially deteriorate the standard deviation of the phase error even without providing much useful information. Figure 3.25 illustrates this concept. For instance, if the radar return signal has a magnitude of 100, an error of magnitude 1 will result in a phase error of up to 0.01 rad. However, if the magnitude of the return signal is 1, an error of 1 can result in much larger phase errors.

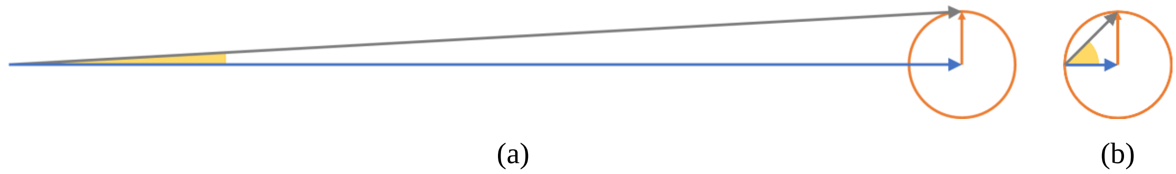


Figure 3.25 – The effect of magnitude over the phase error: (a) strong return signal, (b) return signal close to the noise level. The signal without error is in blue; the random error is in orange; the signal with error is in gray; the resulting phase error is in yellow.

In order to prevent this, an SNR threshold was chosen for the analyses of the following subsections. This subsection explains this choice using a 2D and a 3D setup from Table 3.7, namely the setups that produced the highest phase error of the case study: $L = 5$ with the first split into image blocks of (8 × 4 × 1).

3.4.2.1 Output images

Figure 3.26 and Figure 3.27 present the 2D output images processed by the BP and the FFBP algorithms. Note that both figures clearly show the tree lines of the eucalyptus plantation and that, qualitatively, they are very similar. Indeed, the degree of coherence between them was 0.9942; the magnitude error had a -0.2 dB mean and a 2.3 dB standard deviation; the mean phase error was 0.0004 rad (0.02°); and the standard deviation of the phase error was 0.33 rad (18.8°), somewhat below π/8 rad.

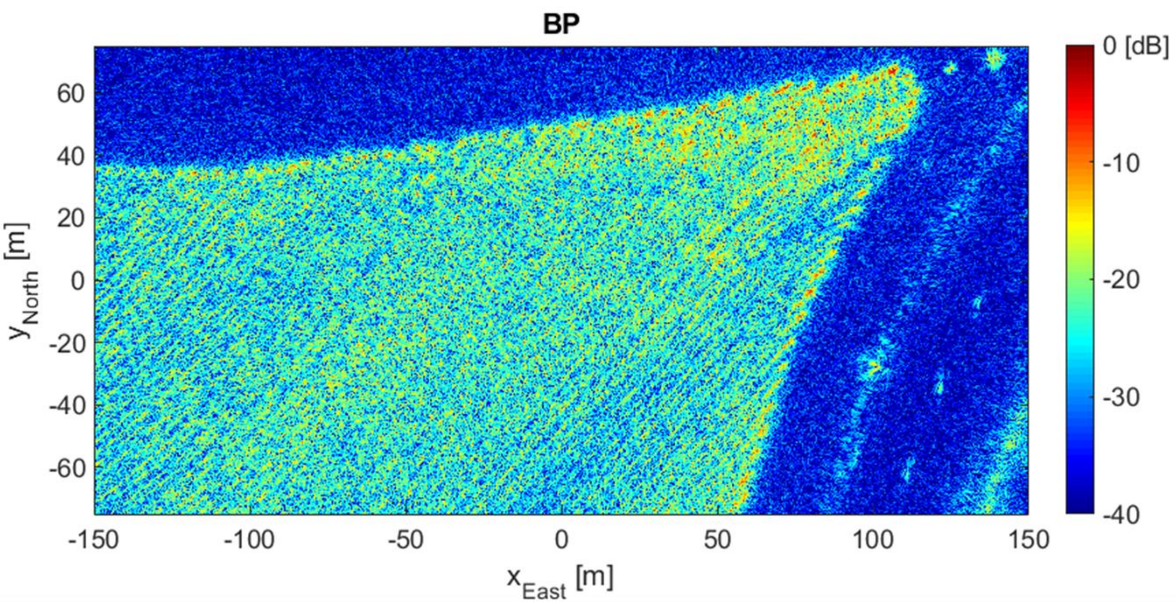


Figure 3.26 – 2D output image processed by the BP algorithm. Normalized magnitude in dB. Source: Góes *et al.*, 2021 [14].

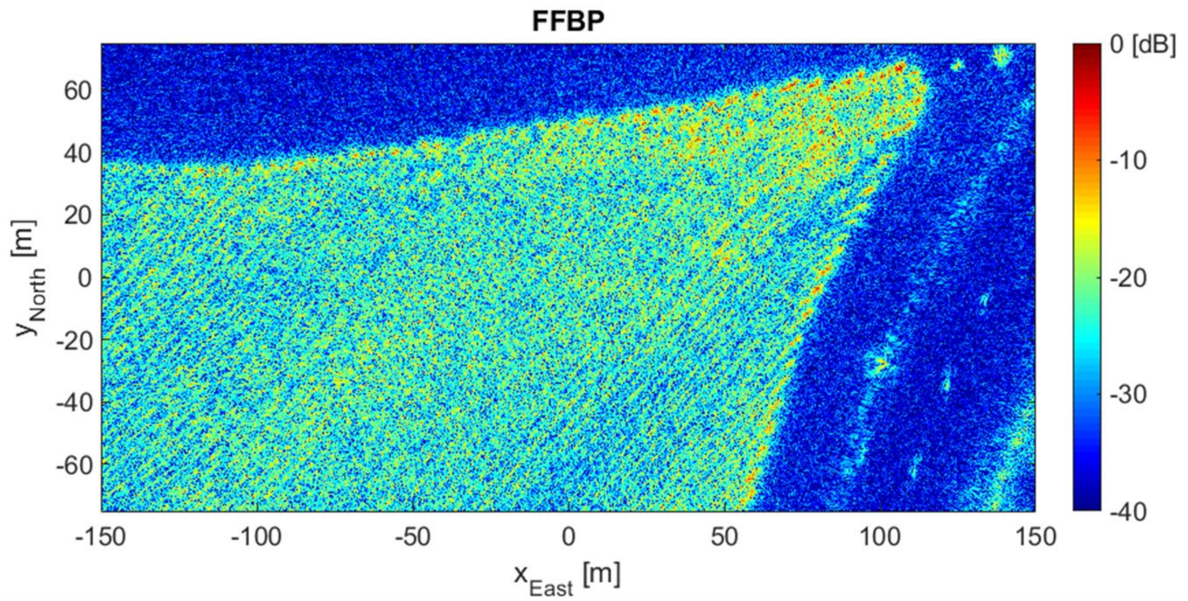


Figure 3.27 – 2D output image processed by the FFBP algorithm with $L = 5$ and a first split into image blocks of (8 × 4 × 1). Normalized magnitude in dB. Source: Góes *et al.*, 2021 [14].

Figure 3.28 and Figure 3.29 show the 3D output images processed by the BP and the FFBP algorithms. They depict isosurfaces at -15 dB normalized magnitude, clearly showing that the radar detects each eucalyptus tree. Again, qualitatively, the differences between the two images are barely perceptible. The degree of coherence between them was 0.9916; the magnitude error had a -0.3 dB mean and a 2.5 dB standard deviation; the mean phase error was 0.0007 rad (0.04°); and the standard deviation of the phase error was 0.35 rad (19.9°), also below π/8 rad.

Figure 3.26 to Figure 3.29 show processed images from data acquired with a spiral flight path. If the same area were surveyed with a linear flight path, the resulting image would have a slant range resolution of 2.4 m (see Table 3.6). It would also have an azimuth resolution of 36 cm, calculated from the wavelength and the horizontal beamwidth (2.22). However, thanks to the 360° acquisition, the resolution might be better than 36 cm across all directions in the (x, y) plane. Indeed, the best attainable resolution in the (x, y) plane would be a quarter of a wavelength (18 cm) [35], [43], [44].

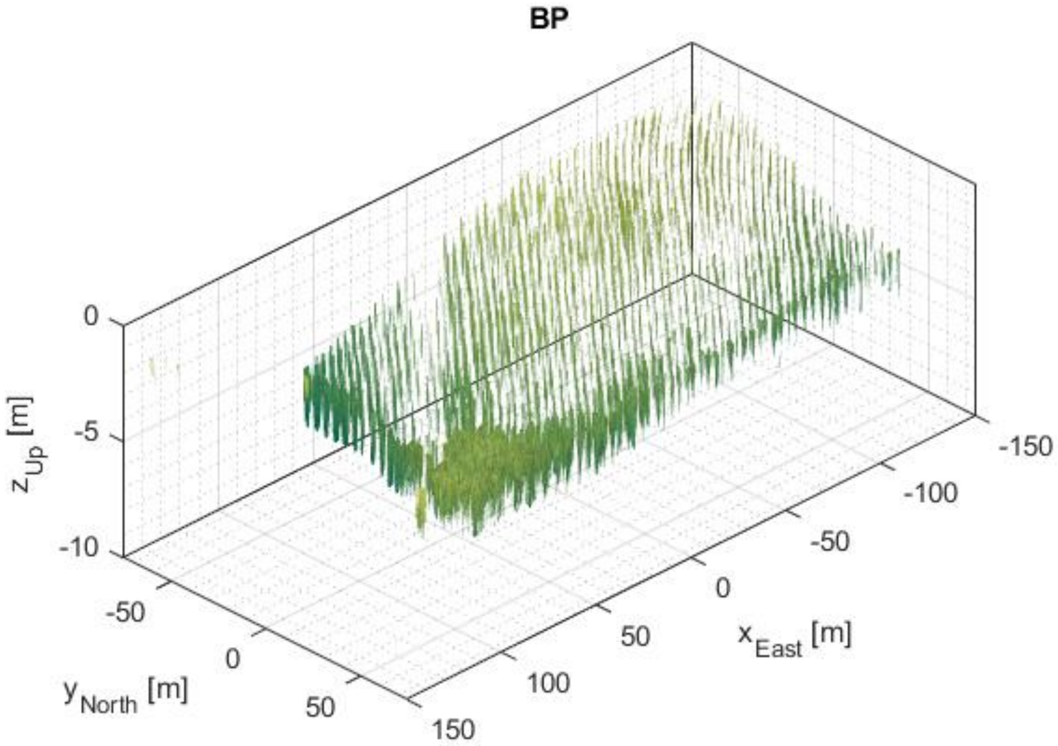


Figure 3.28 – 3D output image processed by BP algorithm. Perspective view of the isosurfaces at -15 dB normalized magnitude. Source: Góes *et al.*, 2021 [14].

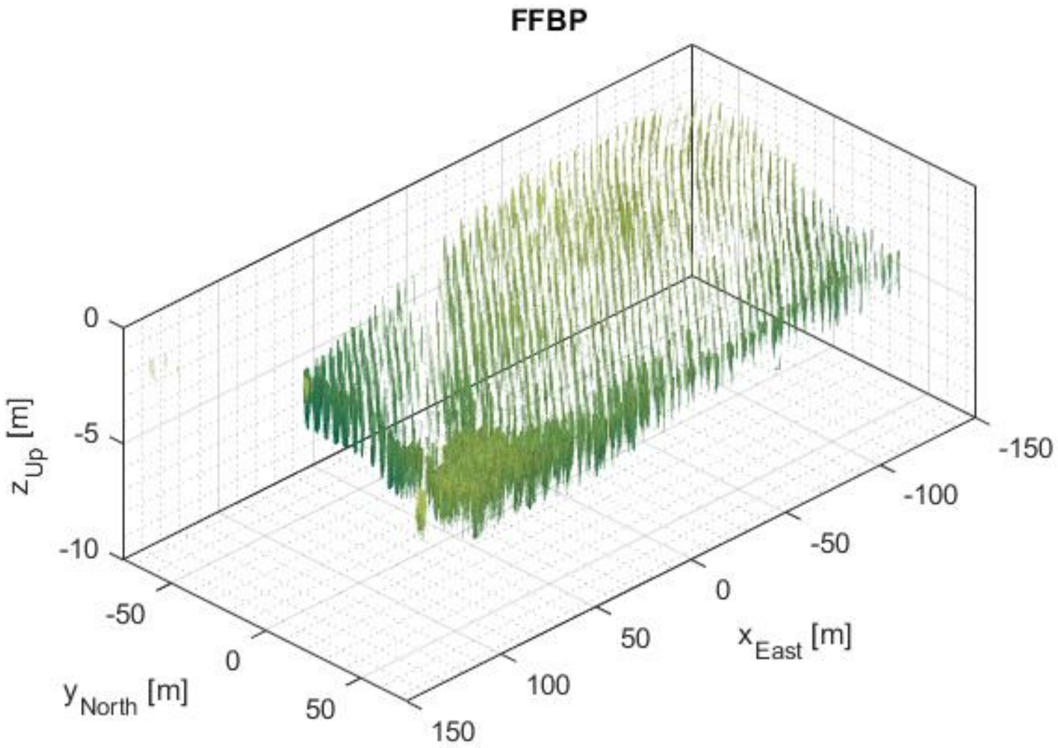


Figure 3.29 – 3D output image processed by the FFBP algorithm with $L = 5$ and a first split into image blocks of $(8 \times 4 \times 1)$. Perspective view of the isosurfaces at -15 dB normalized magnitude. Source: Góes *et al.*, 2021 [14].

3.4.2.2 Comparing algorithms

Figure 3.30 presents the phase error response between the 2D images shown in Figure 3.26 and Figure 3.27. Note that the darkest area of Figure 3.26 corresponds to an increase in phase error in Figure 3.30, which is a sign of noisy behavior. In the northwest corner of Figure 3.26, the mean normalized magnitude at a 30 × 30 m^2 square was close to -40 dB. Thus, this value was considered the noise floor level for calculating the signal-to-noise ratio (SNR) for the upcoming analyses.

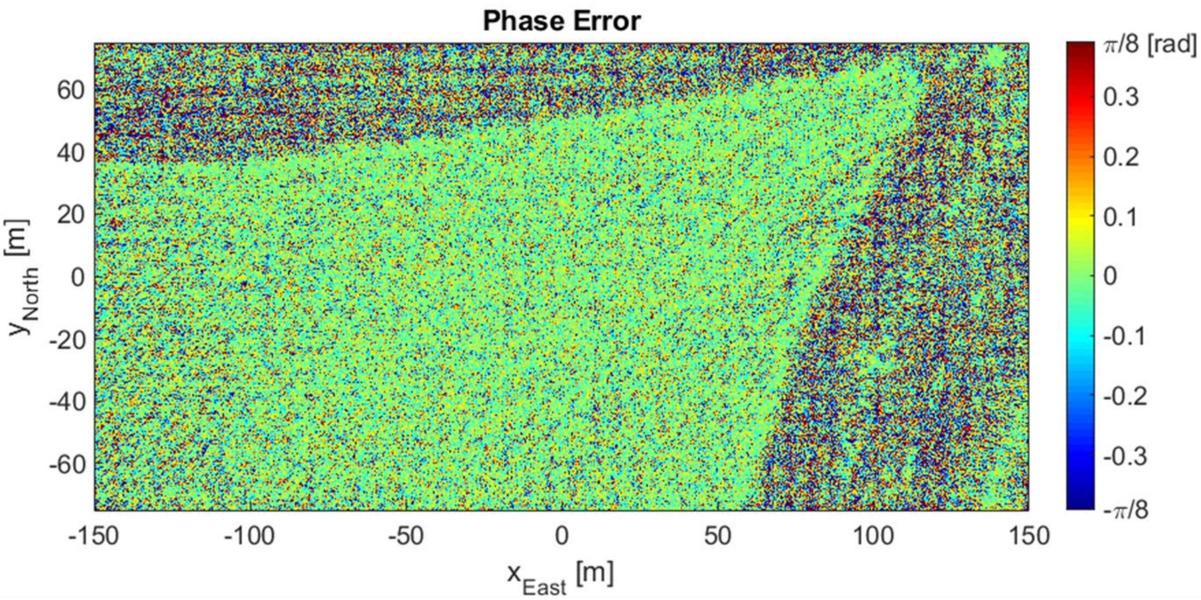


Figure 3.30 – Phase error for the 2D FFBP image with $L = 5$ and an (8 × 4 × 1) first split. Source: Góes *et al.*, 2021 [14].

Figure 3.31 shows the difference between the 3D images of Figure 3.28 and Figure 3.29, reproducing isosurfaces at -15 dB normalized magnitude. The differences were more pronounced when the return signal was close to the noise level, corresponding to the dark blue area in Figure 3.26. Thus, the northern and eastern boundaries are more prominent in Figure 3.31. As a result, the DEM height variations are easy to see on these boundaries.

3.4.2.3 SNR threshold

Figure 3.32 displays three histograms for the 2D phase error in Figure 3.30: (a) with no SNR threshold, i.e., all pixels were taken into account; (b) with a 0 dB SNR threshold; and (c) with a 10 dB SNR threshold. The change was subtle between Figure 3.32(a) and Figure 3.32(b), less than 0.02 for each bin. However, the standard deviation of the phase error decreased from 0.33 rad (18.8°) to 0.20 rad (11.4°). On the other hand, between Figure 3.32(a) and Figure 3.32(c), the change was more pronounced, greater than 0.06 for the central bins. Thus, the standard deviation of the phase error decreased even more to 0.10 rad (5.9°).

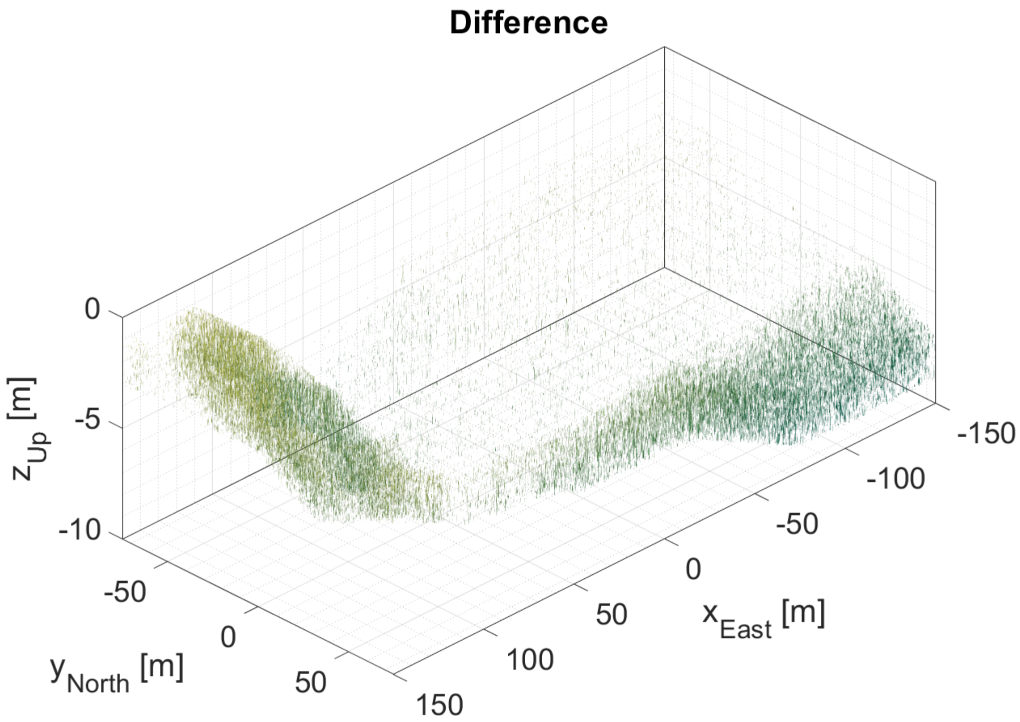
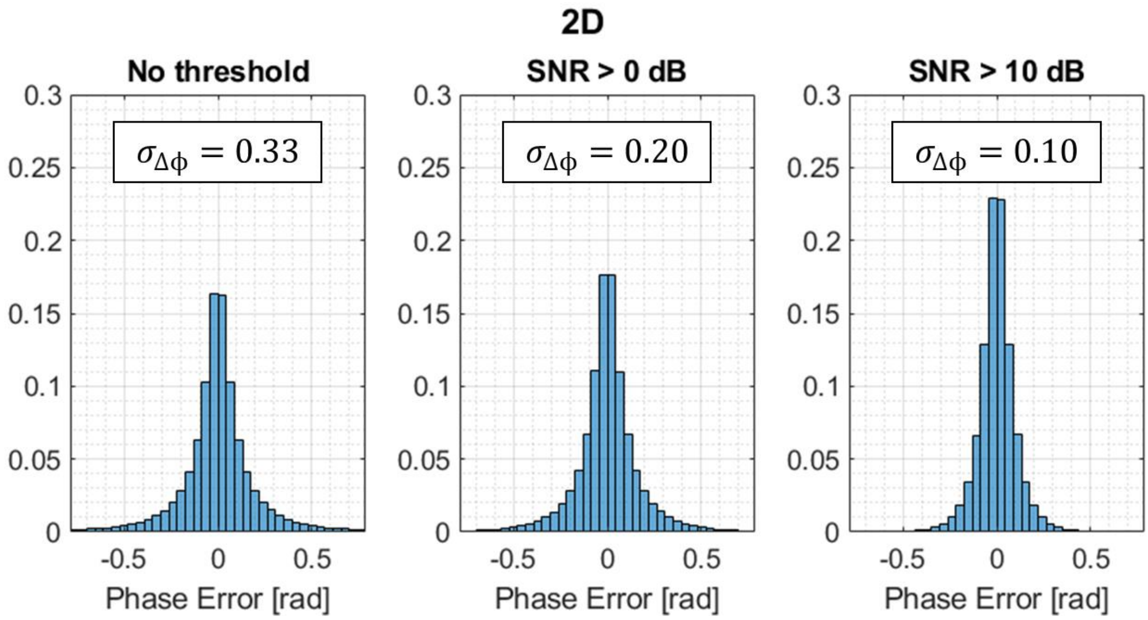


Figure 3.31 – Difference between the 3D BP image and the 3D FFBP image processed with $L = 5$ and an (8 × 4 × 1) first split. Perspective view of the isosurfaces at -15 dB normalized magnitude.



(a) (b) (c)

Figure 3.32 – Histogram of the phase error for the 2D FFBP image with $L = 5$ and an (8 × 4 × 1) first split for (a) no signal-to-noise ratio (SNR) threshold, (b) SNR > 0 dB, and (c) SNR > 10 dB. Source: Góes *et al.*, 2021 [14].

Similarly, Figure 3.33 displays three histograms with different SNR thresholds, corresponding to the 3D phase error of Figure 3.29. Once again, from no SNR threshold (Figure 3.33(a)) to a 0 dB threshold (Figure 3.33(b)), the change was subtle and less than 0.02 for each bin. Nevertheless, the standard deviation of the phase error decreased from 0.35 (19.9°) to 0.22 (12.4°) rad. In contrast, from no SNR threshold (Figure 3.33(a)) to a 10 dB threshold (Figure 3.33(c)), the change was more noticeable and greater than 0.06 for the central bins. Hence, the standard deviation of the phase error decreased further to 0.11 rad (6.5°).

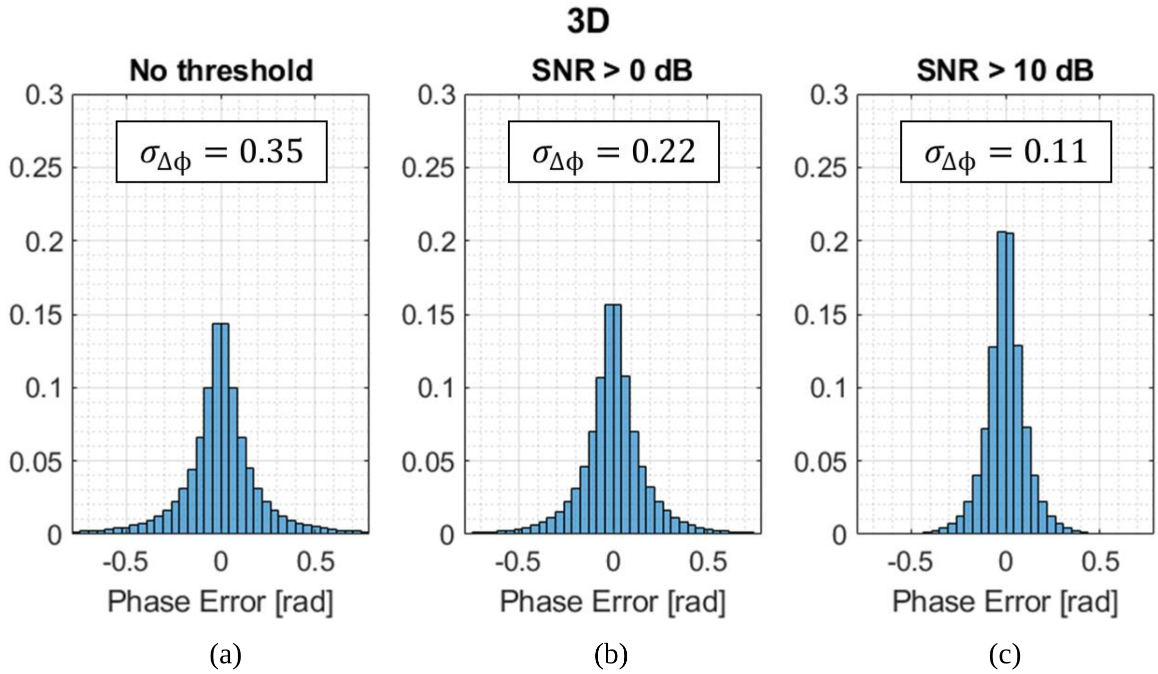


Figure 3.33 – Histogram of the phase error for the 3D FFBP image with $L = 5$ and an (8 × 4 × 1) first split for (a) no SNR threshold, (b) SNR > 0 dB, and (c) SNR > 10 dB.

Table 3.8 further compares the different SNR thresholds. It shows the percentage of pixels and voxels whose phase error is within the ranges ± π/8 and ± π/36 (5°). These ranges corresponded respectively to 18 and 4 bins at the center of the histograms since all bins were π/72 wide. Figure 3.32 and Figure 3.33 only show phase errors within ± π/4 (36 bins), but the total number of bins was 144 for no SNR threshold. Applying a 0 dB threshold did not change the number of bins, but applying a 10 dB threshold resulted in 93 bins for 2D and 129 bins for 3D. Thus, the rate changes in Table 3.8 were less significant for the 0 dB threshold (up to 4.5%) than for the 10 dB threshold (from 8.8 to 18.3%).

As the 10 dB SNR threshold might have eliminated valuable information, the 0 dB SNR threshold was selected for the analyses in the following subsections.

Table 3.8 – Percentage of pixels and voxels for which the phase error is within the indicated ranges.

Image	Phase Error Range	No. Bins	No threshold	SNR > 0 dB	SNR > 10 dB
2D	[-π/8, π/8]	18	90.6 %	94.9%	99.5 %
	[-π/36, π/36]	4	53.2 %	57.3 %	71.5 %
3D	[-π/8, π/8]	18	89.4 %	93.9 %	99.2 %
	[-π/36, π/36]	4	48.7 %	52.7 %	66.8 %

3.4.3 Phase Error vs. Input Parameters

In this subsection, we use linear regression models to verify if we can predict the standard deviation of the phase error ($\sigma_{\Delta\varphi}$) from the term κ, defined in (3.40). To assess the goodness-of-fit of each regression model, we check if they have high coefficients of determination ($\mathcal{R}^2$) and low root-mean-square errors (RMSE). The $\mathcal{R}^2$ measures how well the regression model explains the data; its value is always within the interval [0,1]. Furthermore, to verify if $\sigma_{\Delta\phi} \propto \kappa$, we must test if the linear regression models have a statistically significant slope and a statistically insignificant intercept.

Moreover, we first compare the performances of κ_1, κ_{max} and κ_{avg} to ensure that κ_1 is a good choice for predicting $\sigma_{\Delta\phi}$. Then, we refine the linear regression models.

3.4.3.1 Comparing performances

Figure 3.34 to Figure 3.36 show scatter plots of $\sigma_{\Delta\phi}$ against κ_1, κ_{max} and κ_{avg}, with their respective linear regression models. In addition, Table 3.9 to Table 3.11 present the statistics for these models: the estimate, the standard error (SE), the 95% confidence interval (CI), and the *p*-value for the intercepts and the slopes, as well as the $\mathcal{R}^2$ and RMSE of each regression model. For every intercept, the *p*-value was much higher than 0.05, and the 95% CI crossed zero. Thus, every intercept was statistically insignificant. On the other hand, all slopes had infinitesimal *p*-values, and neither of their CIs crossed zero. Hence, we can reject the null hypothesis of zero slopes. Moreover, all linear regression models presented high $\mathcal{R}^2$ > 0.92 and low RMSE < 1° (0.017 rad). Therefore, the data supported the hypothesis ($\sigma_{\Delta\phi} \propto \kappa$) in all three conditions for calculating κ.

Figure 3.34 and Table 3.9 show the results for the 2D dataset. Note that all data points coincided for κ_1 and κ_{max} and that hence their linear regression models were identical.

In contrast, the κ_1 estimated slope was outside the κ_{avg} CI and vice versa. Therefore, these slopes were significantly different, even if there was a slight overlap between their CIs. The slope was greater for κ_{avg} to compensate for the fact that $\kappa_{avg} < \kappa_1$ for 70% of the data points. Note that, in Figure 3.34, most orange circles (κ_1) and purple dashes (κ_{max}) are to the right of the teal crosses (κ_{avg}).

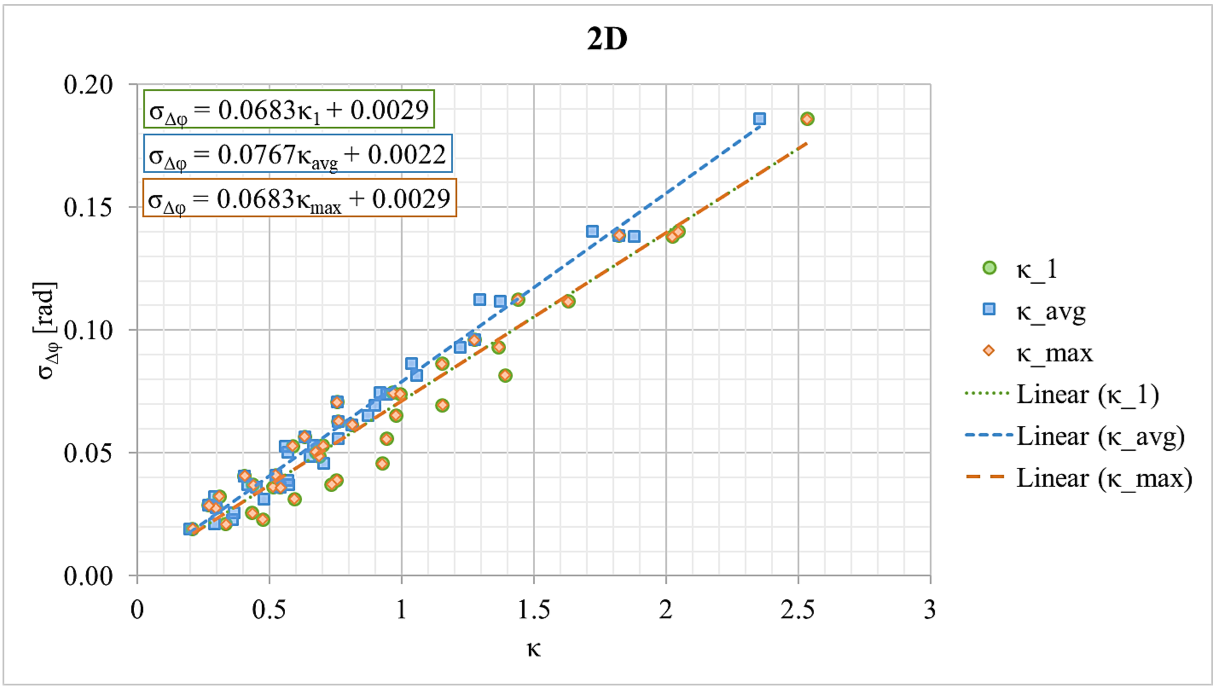


Figure 3.34 – Standard deviation of the phase error ($\sigma_{\Delta\varphi}$) against κ_1, κ_{max} and κ_{avg} for the 2D dataset.

Table 3.9 – Statistics of the linear regression models for the 2D dataset.

κ	Coefficient	Estimate	SE	95% CI	*p*-Value	$\mathcal{R}^2$	RMSE
κ_1	Intercept	0.0029	0.0028	(-0.0029, 0.0086)	0.32	0.943	0.0093
	Slope	0.0683	0.0027	(0.0628, 0.0738)	$\sim 10^{-24}$		
κ_{avg}	Intercept	0.0022	0.0018	(-0.0014, 0.0057)	0.22	0.978	0.0058
	Slope	0.0767	0.0019	(0.0729, 0.0805)	$\sim 10^{-32}$		
κ_{max}	Intercept	0.0029	0.0028	(-0.0029, 0.0086)	0.32	0.943	0.0093
	Slope	0.0683	0.0027	(0.0628, 0.0738)	$\sim 10^{-24}$		

Figure 3.35 and Table 3.10 show the results for the 3D dataset. Now, only 35% of the data points in κ_1 were identical to κ_{max}. Nevertheless, their slopes were statistically equal because the κ_1 CI contained the κ_{max} CI. The κ_1 CI also contained the κ_{avg} estimated slope,

though in this case, the overlap between their CIs was not complete. Still, we cannot affirm that the linear regression models for κ_1 and κ_{avg} were significantly different.

In contrast, the κ_{max} estimated slope was outside the κ_{avg} CI and the other way around. So, like before, these slopes were significantly different, and once again, the κ_{avg} had a greater slope. Furthermore, every data point now had a κ_{avg} (teal cross) lower than the corresponding κ_{max} (purple dash).

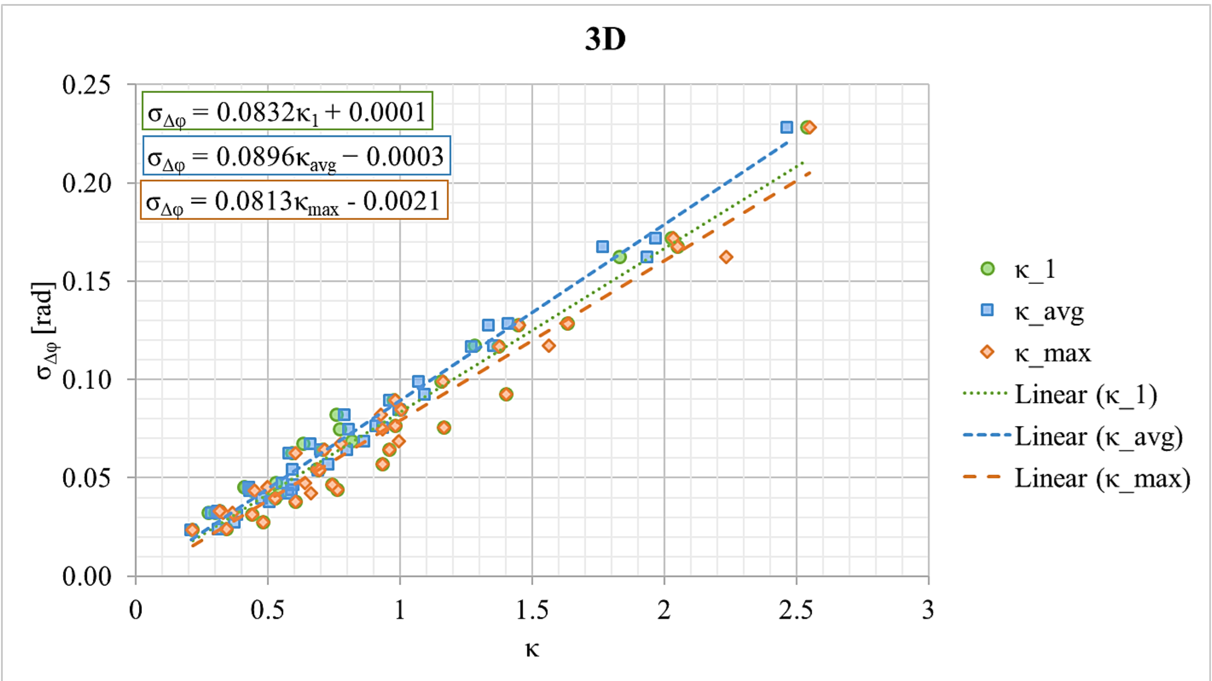


Figure 3.35 – $\sigma_{\Delta\varphi}$ against κ_1, κ_{max} and κ_{avg} for the 3D dataset.

Table 3.10 – Statistics of the linear regression models for the 3D dataset.

κ	Coefficient	Estimate	SE	95% CI	*p*-Value	$\mathcal{R}^2$	RMSE
κ_1	Intercept	0.0001	0.0035	(-0.0069, 0.0071)	0.97	0.943	0.0113
	Slope	0.0832	0.0033	(0.0765, 0.0899)	$\sim 10^{-24}$		
κ_{avg}	Intercept	-0.0003	0.0020	(-0.0044, 0.0038)	0.88	0.980	0.0067
	Slope	0.0896	0.0021	(0.0854, 0.0938)	$\sim 10^{-33}$		
κ_{max}	Intercept	-0.0021	0.0032	(-0.0086, 0.0044)	0.52	0.952	0.0103
	Slope	0.0813	0.0030	(0.0753, 0.0873)	$\sim 10^{-26}$		

Figure 3.36 and Table 3.11 show the results for the combined dataset, comprising both 2D and 3D data. Because the 2D and 3D datasets had the same number of points (40 each), $\kappa_1 = \kappa_{max}$ for 67.5% of the data points. Again, the κ_1 CI contained the κ_{max} CI, so their slopes

were statistically equal. On the other hand, though they overlapped to some extent, neither the κ_1 CI contained the κ_{avg} estimated slope nor the κ_{avg} CI contained the κ_1 estimated slope. Thus the κ_1 and κ_{avg} slopes were significantly different. The κ_{avg} slope was also different from the κ_{max} slope because their CIs did not overlap. As before, κ_{avg} had the smallest values for most data points, so its linear regression model had the highest slope.

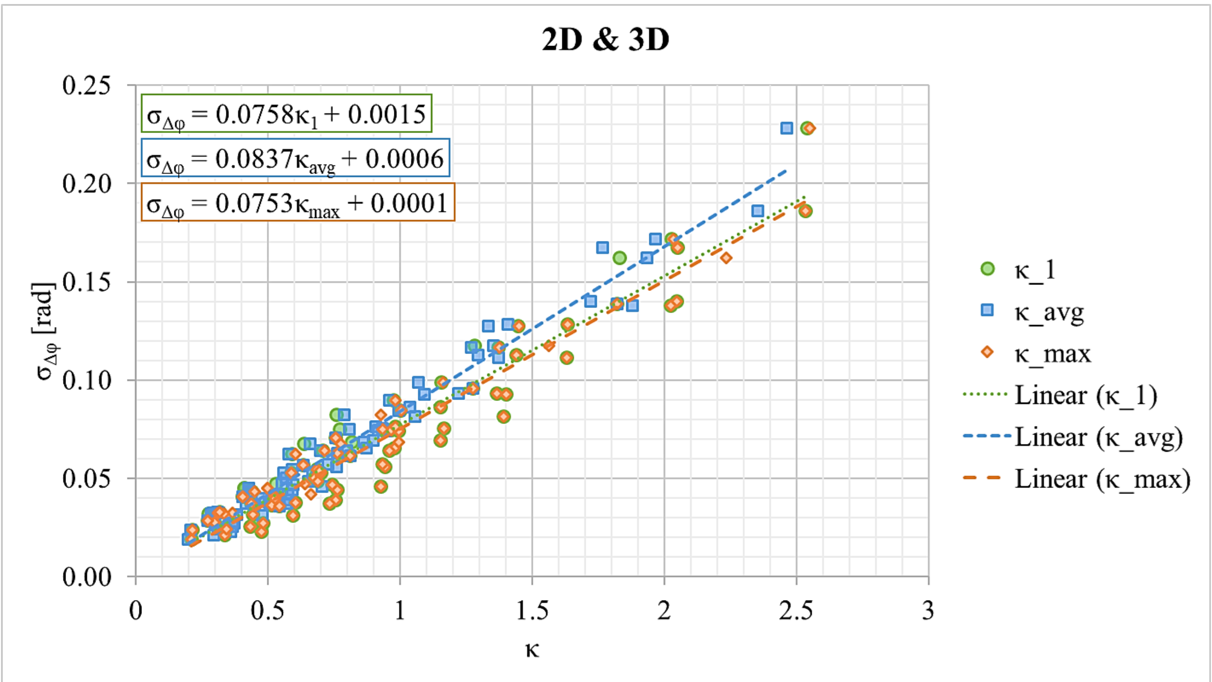


Figure 3.36 – $\sigma_{\Delta\varphi}$ against κ_1, κ_{max} and κ_{avg} for the combined (2D & 3D) dataset.

Table 3.11 – Statistics of the linear regression models for the combined (2D & 3D) dataset.

κ	Coefficient	Estimate	SE	95% CI	*p*-Value	$\mathcal{R}^2$	RMSE
κ_1	Intercept	0.0015	0.0026	(-0.0038, 0.0067)	0.58	0.920	0.0122
	Slope	0.0758	0.0025	(0.0708, 0.0809)	$\sim 10^{-44}$		
κ_{avg}	Intercept	0.0006	0.0017	(-0.0029, 0.0041)	0.73	0.965	0.0081
	Slope	0.0837	0.0018	(0.0801, 0.0873)	$\sim 10^{-58}$		
κ_{max}	Intercept	0.0001	0.0024	(-0.0046, 0.0049)	0.96	0.936	0.0109
	Slope	0.0753	0.0022	(0.0708, 0.0797)	$\sim 10^{-47}$		

For all datasets, the linear regression models of κ_1 and κ_{max} were statistically equivalent. These results can be explained by how the splitting function works. After the first split into image blocks and before the last recursion, the function makes the number of divisions $D_x = D_y = L$; but at the last recursion step, it makes $D_x > L$ and $D_y > L$. Moreover, after the number of voxels in the z-direction reaches (or surpasses) the expected value, the function keeps $D_z = 1$ for the remaining recursions.

Ideally, the splitting function should divide the image by L in each direction for all recursions to keep the product $\delta_k \Delta_c$ constant. However, that could lead to waste, as the algorithm would have to process a far larger image than required. So, as a compromise, the function tries to prevent the product $\delta_k \Delta_c$ from increasing to avoid deteriorating the phase error. This strategy worked well for the 2D dataset, to the point that all data points coincided for κ_1 and κ_{max}.

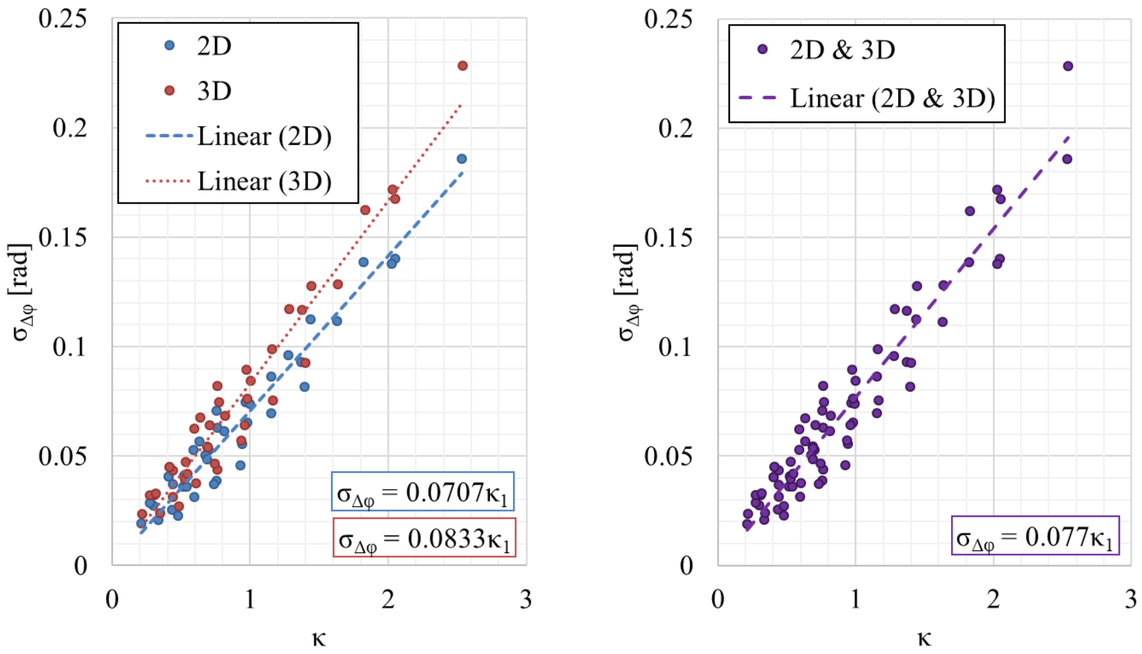
The linear regression models that best fit the data were those for κ_{avg}. Indeed, the κ_{avg} linear regression models had the highest $\mathcal{R}^2$ and the lowest RMSE of each dataset. This result aligns with the error analysis in [31], where the phase error correlates to the average range error across all subapertures and recursions. However, κ_{avg} is more difficult to determine *a priori* than κ_1. While κ_1 can be calculated at the start of processing, κ_{avg} (and κ_{max}) can only be calculated after generating the splitting scheme. Furthermore, the linear regression models for κ_1 also have good fits for all datasets. Therefore, we can use setup parameters to calculate κ_1 and then predict $\sigma_{\Delta\phi}$.

3.4.3.2 Refining the regression models

For all linear regression models above, the intercepts were statistically insignificant. Therefore, new linear regression models were created with zero intercepts to test hypothesis (3.40) again. However, this process was only conducted for κ_1.

Figure 3.37 presents scatter plots of $\sigma_{\Delta\phi}$ against κ_1. Figure 3.37(a) shows two separate linear regressions for the 2D and 3D datasets; Figure 3.37(b) shows a single linear regression for the combined dataset. In Figure 3.37(a), the linear regression of the 3D dataset had a slightly steeper slope, indicating higher phase errors than for the 2D dataset. The reason is that the number of voxels in the x- and y-directions was significantly larger than in the z-direction. Thus, the volumetric images were only split across the x- and y-directions for the last recursions in most setups. However, according to the method for controlling the phase error proposed in [31], the splitting scheme should keep the product $\delta_k \Delta_c$ constant across all recursion steps. Therefore, this balance was possible when processing the 2D images but not the 3D images.

Table 3.12 shows the statistics for all three linear regression models. All slopes had meager p-values, and none of their CIs crossed zero. Also, all linear regression models presented high $\mathcal{R}^2 > 0.92$ and low RMSE ≤ 0.0122 rad (0.7°). Therefore, hypothesis (3.40) was validated once again.



(a) (b)

Figure 3.37 – $\sigma_{\Delta\varphi}$ against κ_1, with (a) separate linear regressions and (b) a single linear regression for 2D and 3D data.

Table 3.12 – Statistics of the linear regression models with zero intercepts for κ_1.

Data	Estimate	SE	95% CI	p-Value	$\mathcal{R}^2$	RMSE
2D	0.0707	0.0014	(0.0678, 0.0735)	~10^{-36}	0.945	0.0093
3D	0.0833	0.0017	(0.0799, 0.0867)	~10^{-36}	0.943	0.0112
2D & 3D	0.0770	0.0013	(0.0744, 0.0796)	~10^{-66}	0.922	0.0122

3.4.4 Phase Error vs. Processing Time

In this subsection, we evaluate the performance of the FFBP algorithm in terms of processing time. Some results presented here were extracted from the same datasets of the previous subsection. However, instead of calculating $\sigma_{\Delta\phi}$ for two selected image blocks (Figure 3.24), $\sigma_{\Delta\phi}$ was calculated for the whole image.

Figure 3.38 presents $\sigma_{\Delta\phi}$ against processing time at different values of L for the 2D dataset analyzed in the last subsection. Note that the smaller the phase error, the greater the processing time and vice versa, confirming the trade-off identified in [30].

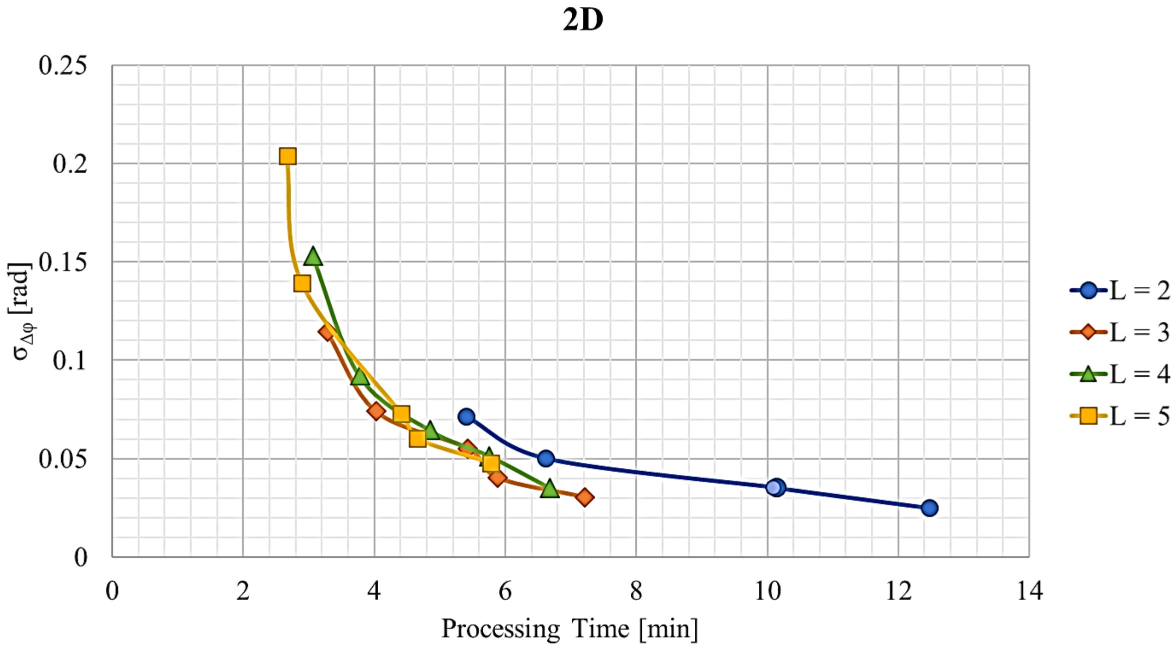


Figure 3.38 – $\sigma_{\Delta\phi}$ against processing time at different values of L for the 2D dataset. Source: Góes *et al.*, 2021 [14].

It is clear that the curve for $L = 2$ is far slower than the others, but at closer inspection, we can see that all curves move left and upwards as L increases. They behave this way because the highest the value of L, the longest the subaperture (δ_k). However, there was also another factor influencing the phase error. Though the first splits given at the input were the same, the actual values varied with L. As L increased, the number of image blocks often decreased, thus producing longer subimage diagonals (Δ_c). Table 3.13 shows the actual first split into image blocks as a function of L. The input first splits were repeated for convenience.

Table 3.13 – First split into image blocks at the input and the actual first split as a function of L.

	L	**First split into image blocks**				
Input value		8 × 4 × 1	12 × 6 × 1	16 × 8 × 1	20 × 10 × 1	24 × 12 × 1
Actual value	2	12 × 6 × 1	16 × 8 × 1	24 × 12 × 1	24 × 12 × 1	32 × 16 × 1
	3	10 × 5 × 1	14 × 7 × 1	19 × 10 × 1	21 × 11 × 1	28 × 14 × 1
	4	9 × 5 × 1	14 × 7 × 1	19 × 10 × 1	24 × 12 × 1	25 × 13 × 1
	5	9 × 5 × 1	12 × 6 × 1	17 × 9 × 1	20 × 10 × 1	25 × 13 × 1

In Table 3.13, note that $L = 2$ had the same actual first split for two different inputs: $(16 \times 8 \times 1)$ and $(20 \times 10 \times 1)$. As a result, their outputs had virtually the same performance in both processing time and phase error. Figure 3.38 shows one of the markers with a lighter shade of blue to highlight this issue. Table 3.13 is valid for both the 2D and the 3D datasets.

Figure 3.39 shows $\sigma_{\Delta\phi}$ against processing time for the 3D dataset of the previous subsection. Concerning how the value of L affects the phase error, the same reasoning can be applied here. However, note that the curves in Figure 3.39 are not as smooth as in Figure 3.38. The reason is that the splitting function causes unnecessary waste and needs improvement.

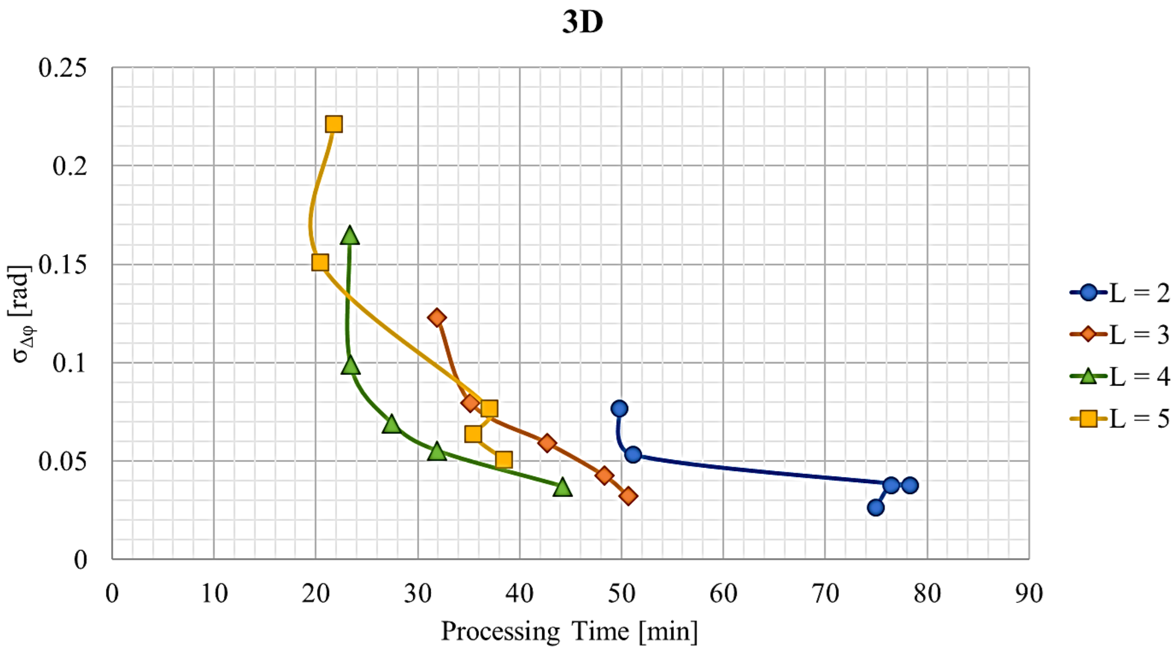
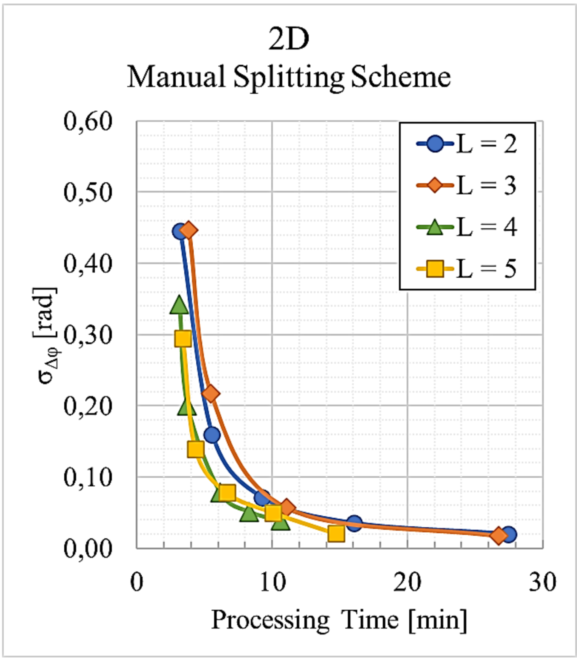


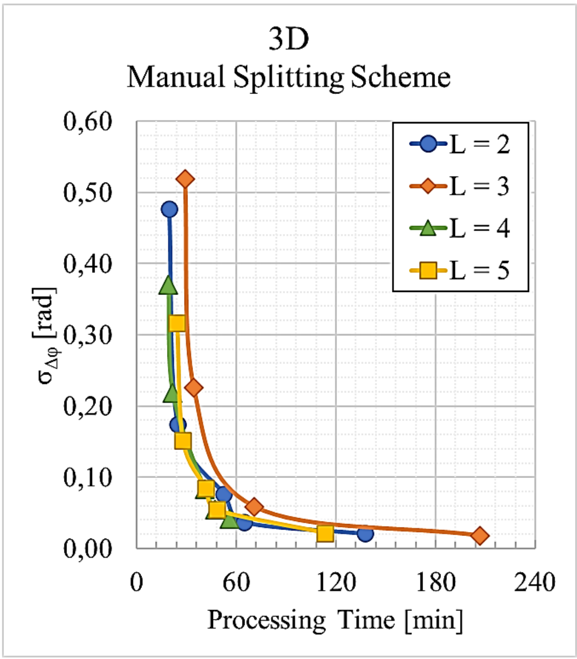
Figure 3.39 – $\sigma_{\Delta\phi}$ against processing time at different values of L for the 3D dataset. Source: Góes *et al.*, 2021 [14].

If we define surplus as the percentage of pixels or voxels that are disregarded for exceeding the expected value, then the average surplus was 7.4% for the 2D dataset and 36.3% for the 3D dataset. This surplus increase can be explained by the excess of voxels in the *z*-direction. Only the splitting schemes with $L = 4$ produced the expected number of voxels in the *z*-direction (2.4 m/0.2 m = 12). That is why $L = 4$ had the most well-behaved curve in Figure 3.39. For $L = 2$, 3 and 5, the number of voxels in the *z*-direction was 16, 18, and 15, respectively.

Figure 3.40 shows $\sigma_{\Delta\phi}$ against processing time for two additional datasets obtained using manual splitting schemes. Note that both the 2D and 3D datasets produced smooth curves, thus confirming that the splitting function caused the odd behavior seen in Figure 3.39. Moreover, Figure 3.40 provides further detail into the trade-off between the phase error and processing time. If the number of image blocks is excessive, the processing time goes up without causing a significant change in phase error. On the other hand, if the number of image blocks is insufficient, the phase error is intensified, but the processing time scarcely improves. This dynamic is not so apparent in Figure 3.38 and Figure 3.39 because $\sigma_{\Delta\phi}$ was kept below 0.25 rad (14.3°), and the processing time was always under 14 min for the 2D dataset and 90 min for the 3D dataset.



(a)



(b)

Figure 3.40 – $\sigma_{\Delta\phi}$ against processing time at different values of L for the 2D and 3D datasets obtained with manual splitting schemes.

Table 3.14 extracts some results shown in Figure 3.38 and Figure 3.39, listing the slowest, fastest, and average processing times of the FFBP algorithm. Table 3.14 also compares these processing times to the BP algorithm, showing the corresponding speed-up factors. As expected, the speed-up factor was higher for the 3D dataset than for the 2D dataset. From Section 3.1.1, FFBP algorithms can reduce the computing cost from $\mathcal{O}(P^3)$ to $\mathcal{O}(P^2 \log P)$ for 2D images and from $\mathcal{O}(P^4)$ to $\mathcal{O}(P^3)$ for 3D images.

Table 3.14 – Processing time of the slowest, fastest, and average FFBP configurations compared to the BP algorithm. Source: Góes et al., 2021 [14].

Image Type	BP Processing Time	FFBP		
		Configuration	Processing Time	Speed-Up Factor
2D	35 min 33 s	Fastest	2 min 40 s	13.33
		Average	5 min 45 s	6.18
		Slowest	12 min 28 s	2.85
3D	7 h 12 min 18 s	Fastest	20 min 24 s	21.2
		Average	42 min 8 s	10.3
		Slowest	1 h 18 min 18 s	5.52

3.4.5 Discussion

Hypothesis (3.40) was successfully validated for the P-band data as the parameters at the start of processing truly can predict $\sigma_{\Delta\varphi}$ at the output. The hypothesis was also validated when combining the 2D and 3D datasets (Figure 3.37(b)), reinforcing that what matters most for this FFBP algorithm is the diagonal of the subimages, not their width.

Hypothesis (3.40) was inspired by the range error analysis presented in [31]. However, it disregarded the effect of deviations from a linear flight path, as the phase compensation term (3.24) ensures good focusing quality for any flight path. This term was proposed in [39] but with a different goal: to avoid taking range samples at each recursion to accelerate processing.

If (3.24) was removed from the FFBP algorithm, the outcome of this case study would be completely unsatisfactory. Indeed, Figure 3.41 shows the resultant 2D image with $L = 2$ and a $(24 \times 12 \times 1)$ first split, i.e., the configuration with the lowest phase error standard deviation of all datasets. If Figure 3.41 is compared to the BP output image of Figure 3.26, the degree of coherence is a meager 0.12.

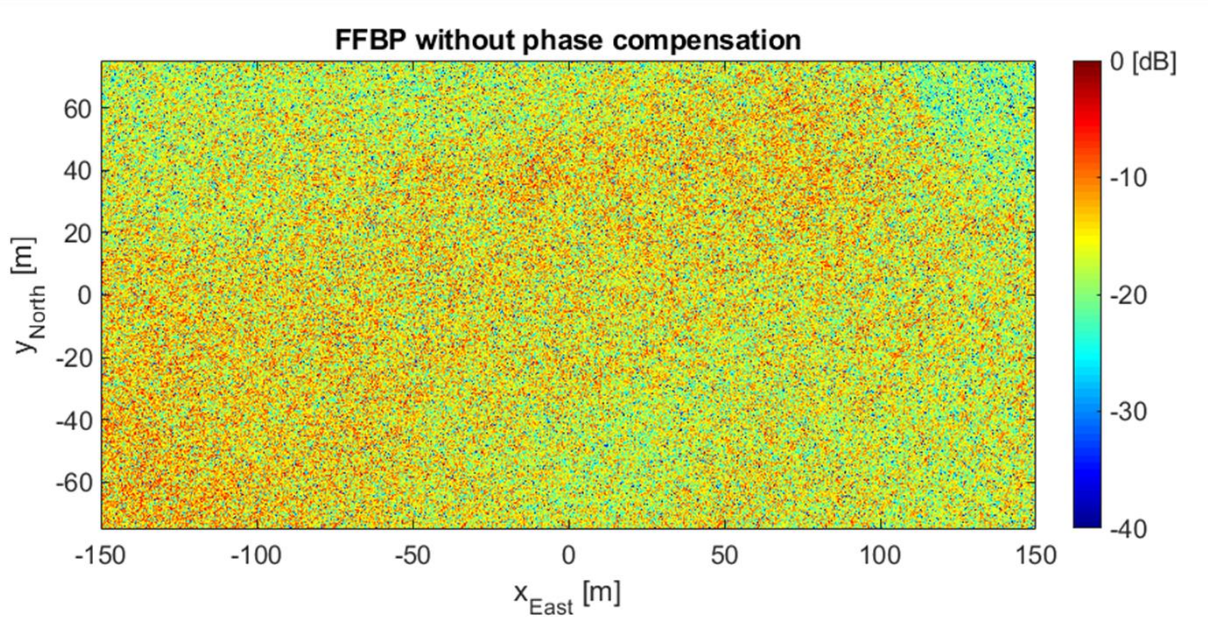


Figure 3.41 – 2D output image processed by the FFBP algorithm without the phase compensation term (3.24) for the setup with $L = 2$ and a (24 × 12 × 1) first split. Normalized magnitude in dB. Source: Góes *et al.*, 2021 [14].

The configuration with the lowest image quality (see Figure 3.27 and Figure 3.29) had the longest subaperture length and subimage diagonal, i.e., $L = 5$ with an (8 × 4 × 1) first split. Likewise, the configuration with the highest image quality had the shortest subaperture length and subimage diagonal, i.e., $L = 2$ with a (24 × 12 × 1) first split. Table 3.15 lists some figures of merit at these extremes for the 2D and 3D datasets, namely the phase error standard deviation, the degree of coherence, and an SNR of equivalent thermal noise. SNR of equivalent thermal noise can be understood as the signal-to-thermal noise ratio that would result in an interferometric image with the same degree of coherence. It is calculated according to [45]. Table 3.15 also shows the values for an average image quality, which corresponds to the following configurations:

- $L = 5$ with a (16 × 8 × 1) first split for 2D;
- $L = 2$ with an (8 × 4 × 1) first split for 3D.

It is important to note that "lowest quality" refers to a relative comparison within the dataset, not to poor quality in absolute terms. Qualitatively, Figure 3.27 and Figure 3.29 appear almost identical to Figure 3.26 and Figure 3.28, indicating that this image quality level is suitable for SAR processing. Indeed, in [9], the same drone-borne SAR system produced a high-accuracy forest inventory with SAR interferometry in the P-band. A 5% accuracy was possible thanks to the forest SNR being higher than 17 dB. Because the SNR of equivalent noise was more than 20 dB, the configurations with the lowest image quality were already satisfactory. Moreover, they were also associated with the fastest processing times (see Table 3.14), with speed-up factors of 13 and 21 for 2D and 3D images, respectively.

Table 3.15 – Performance of the configurations with highest, average, and lowest image quality. Source: Góes *et al.*, 2021 [14].

Dataset	**Figure of Merit**	**Image Quality**		
		Highest	**Average**	**Lowest**
2D	Phase Error Standard deviation	0.025 rad (1.4°)	0.073 rad (4.2°)	0.20 (11.7°)
	Degree of coherence	0.9999	0.9993	0.9945
	SNR of equivalent Thermal noise	40 dB	31 dB	23 dB
3D	Phase Error Standard deviation	0.026 (1.5°)	0.077 rad (4.4°)	0.22 rad (12.7°)
	Degree of coherence	0.9999	0.9988	0.9921
	SNR of equivalent Thermal noise	38 dB	29 dB	21 dB

On the other hand, the configurations with the highest image quality had unnecessarily slow processing times. For example, suppose a specific application would require an SNR higher than 20 dB. In that case, a configuration with average image quality could be employed. The average phase error standard deviation points were close to those with average processing time in Figure 3.38 and Figure 3.39. Therefore, more demanding applications could benefit from a speed-up factor of about 6 for 2D and 10 for 3D images.

4 SPIRAL SAR

The SAR acquisition geometry directly impacts the performance of the output image. For example, high-resolution 3D SAR images can be obtained with multi-circular or spiral flight paths. Typically, these state-of-the-art acquisition geometries have constant radii due to practical reasons. Though thankfully, these limitations would not apply to a drone-borne SAR system.

This chapter proposes a new acquisition geometry that can perform better than the state-of-the-art, consisting of spiral flight paths with variable radii: conical spiral flight paths. In addition, this chapter proposes a methodology for designing this type of flight path. Such a methodology relies on an analytical expression for the vertical resolution. However, the expression found in the literature is unfortunately inaccurate. Therefore, this chapter also proposes a new, revised expression.

Section 4.1 reviews different SAR acquisition geometries found in the literature. Section 4.2 explains a key concept called the wavenumber shift and develops some analytical expressions, including the vertical resolution. Section 4.3 validates that expression using simulation results and investigates the influence of different parameters of spiral flight paths. Finally, Section 4.4 proposes a methodology for designing spiral flight paths and validates it with actual SAR data.

4.1 SAR ACQUISITION GEOMETRY

As seen in Chapter 2, a radar system measures the delay of the return signal. When we integrate this information from different azimuth positions, we enhance the resolution in that direction. That is the principle of SAR.

The SAR acquisition geometry can affect the resolution of the output image. Recall, from Chapter 2, that Stripmap SAR produces images with a finer azimuth resolution than slant range resolution. The first is proportional to the signal wavelength, while the second is inversely proportional to the bandwidth. However, we can obtain high-resolution 2D images with wavelength-dependent resolutions in both directions. To that end, another acquisition geometry is required, such as Circular SAR.

Furthermore, the SAR acquisition geometry influences what information we can extract from the scene. For example, Stripmap SAR with only one receiving antenna will be limited to producing 2D images. The reason is that it cannot provide information on the elevation angle since the radar does not measure the direction of the receiving signal. Such information could be used, for instance, to determine the terrain topography. Therefore, to obtain 3D information, we must employ other acquisition geometries, namely SAR Interferometry, SAR Tomography, Multi-Circular SAR, or Spiral SAR.

4.1.1 SAR Interferometry

SAR Interferometry is a highly accurate technique that estimates the phase difference between two or more radar images acquired from different positions or at different times. SAR Interferometry has many applications, including creating topographic maps, monitoring crop growth [11], and measuring ground displacement [12].

Figure 4.1 illustrates the most basic type of SAR Interferometry: two antennas separated by a baseline (b). The distance traveled by the two incoming signals is slightly different. That difference in distance (ΔR) translates into a phase difference ($\Delta\varphi$) between the two images, calculated by the interferogram [46]:

$$\Delta\varphi = \frac{4\pi}{\lambda}\Delta R \tag{4.1}$$

where λ is the signal wavelength. Then, height information (z_0) can be extracted from the interferometric phase. The expression will differ depending on whether the images were processed using frequency-domain [46] or time-domain [23] algorithms.

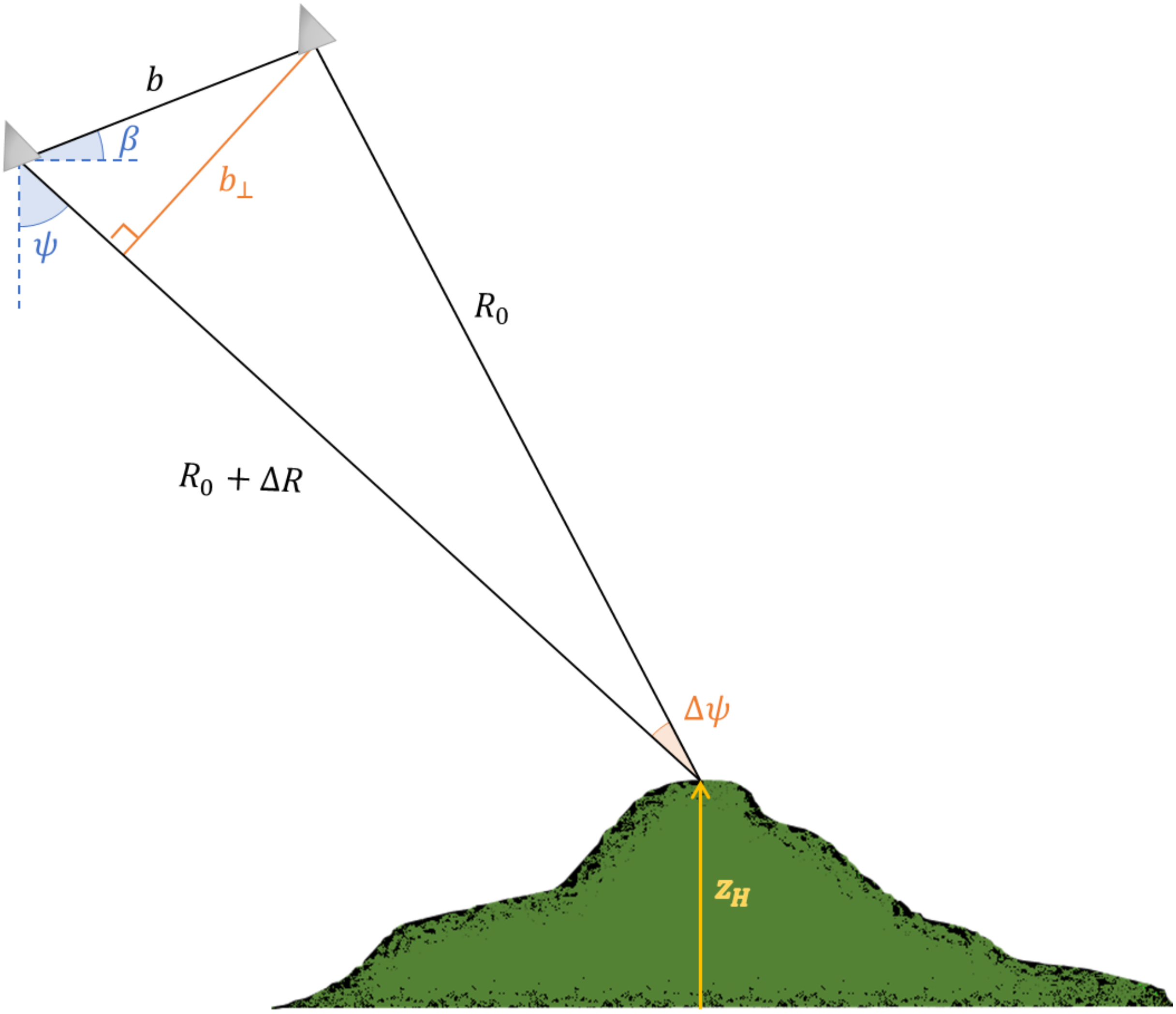


Figure 4.1 – SAR Interferometry: two antennas separated by a baseline (b), generating a difference in distance (ΔR) between the two signal paths, which is used to obtain height information (z_H). Other geometric parameters are the look-angle (ψ), the tilt angle (β), the effective baseline ($b_\perp$) and the difference in look-angle ($\Delta\psi$).

4.1.2 SAR Tomography

SAR Interferometry will not determine height correctly if several targets share the same resolution cell. SAR Tomography presents a solution to this issue: flying several tracks to build another synthetic aperture in elevation (see Figure 4.2), which shall be called the tomographic aperture from now on. Furthermore, in the case of a semi-transparent medium – such as a glacier or a forest – SAR Tomography makes it possible to obtain 3D images [47].

In SAR Tomography, we can determine both the resolution ($\delta_{LOS_\perp}$) and the maximum width of the imaged object ($H_{LOS_\perp}$) perpendicular to the line-of-sight (LOS) direction [48]:

$$\delta_{LOS_\perp} = \frac{\lambda R_0}{2B} \tag{4.2}$$

$$H_{LOS_\perp} \leq \frac{\lambda R_0}{2\Delta B} \tag{4.3}$$

B is the tomographic aperture length, ΔB is the sampling distance, i.e., the distance between tracks, and R_0 is the shortest distance from the radar to the target. These equations were determined in [48] for a vertical tomographic aperture with the same mean height as the target. In practice, though, it is usual to build a slanted tomographic aperture, perpendicular to the LOS direction, as illustrated in Figure 4.2 [47], [49], [50].

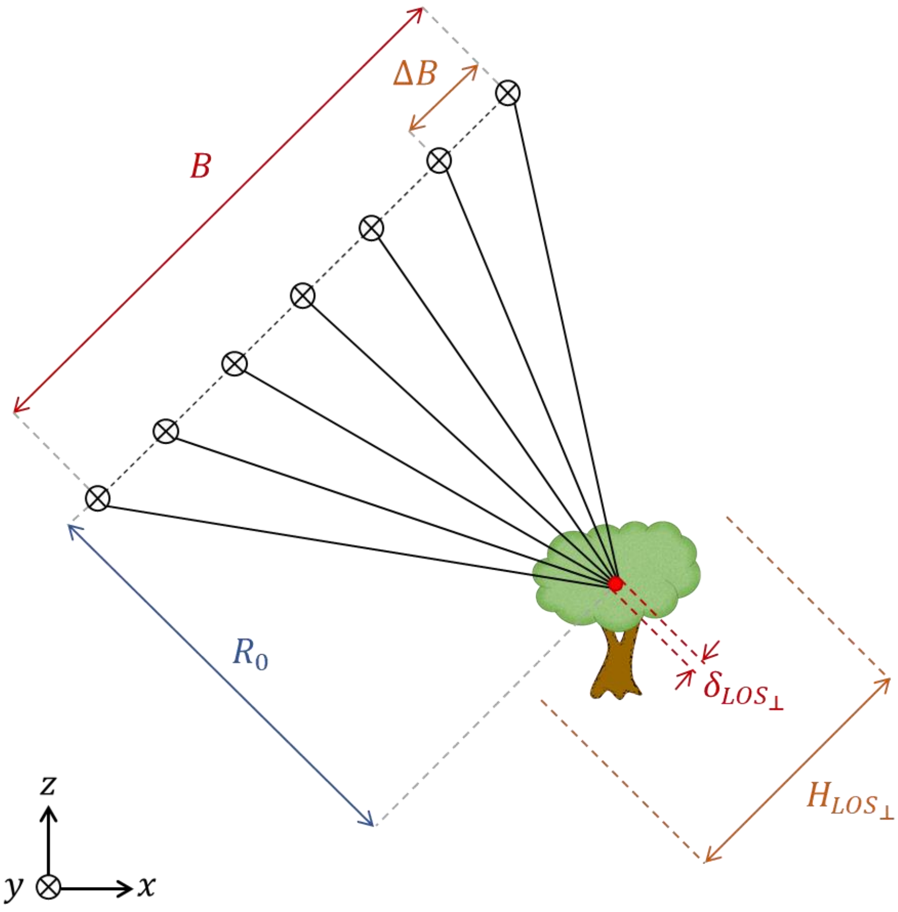


Figure 4.2 – SAR Tomography: several flight tracks going into the page; they are separated by a sampling distance (ΔB), thus building a tomographic aperture in elevation (B). The resolution ($\delta_{LOS_\perp}$) and maximum width ($H_{LOS_\perp}$) perpendicular to the line-of-sight (LOS) direction are functions of the distance from the tomographic aperture to the target (R_0).

4.1.3 Circular SAR

SAR configurations that rely on straight flight paths may become limited when many anisotropic targets (e.g., buildings) are present in the scene. A good alternative is the Circular SAR, which can see the targets from 360°. For this reason, Circular SAR produces images with very high resolution on the ground plane. Furthermore, Circular SAR can resolve altitude ambiguities, thus providing height information (see Figure 4.3) [43].

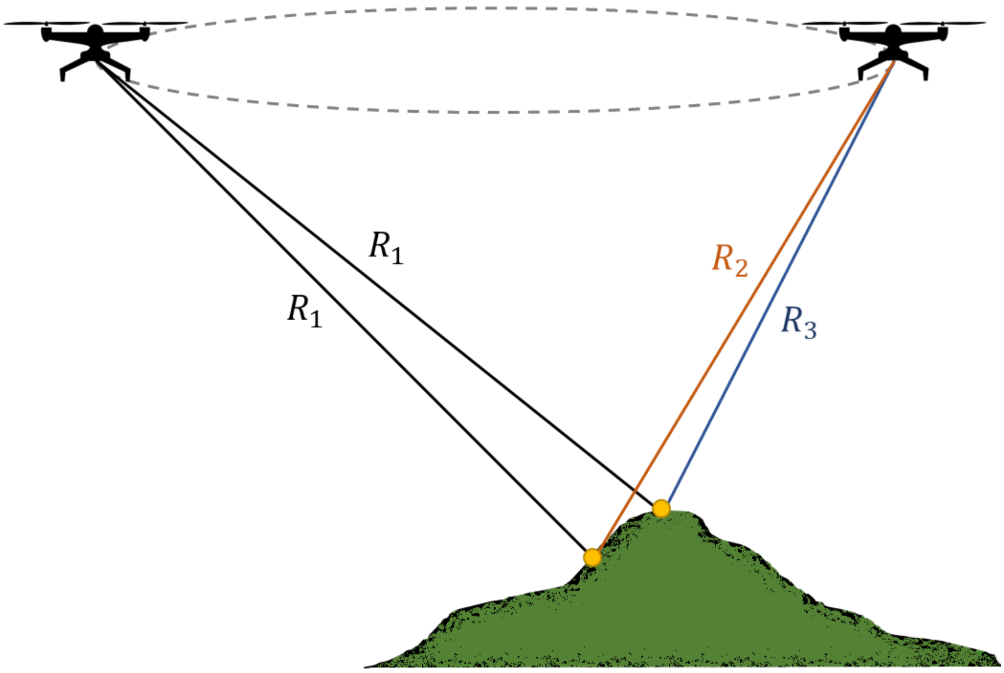


Figure 4.3 – Circular SAR: resolving altitude ambiguity.

Ishimaru *et al*. [43] provided analytical expressions for both the ground resolution (δ_{xy}) and the vertical resolution (δ_z). These expressions were calculated for an isotropic target located at the circular flight path axis (see Figure 4.4). The resolutions are given by [43]:

$$\delta_{xy} = \frac{1.12\lambda}{2\pi \sin(\psi)} \tag{4.4}$$

$$\delta_z = \sqrt{\frac{\ln(2)}{\pi}} \frac{c}{W \cos\psi} \tag{4.5}$$

where ψ is the look-angle. Note that when $\psi = 45°$, $\delta_{xy} \approx \lambda/4$.

According to Ishimaru *et al*. [43], the target's ground response takes the form of a zero-order Bessel function of the first kind, $J_0(u)$. In contrast, the vertical response takes the form e^{-u^2}. Originally, the ground resolution was determined at the first zero of the Bessel function, and the vertical resolution was determined for the point where the amplitude reaches e^{-1} of the maximum value. Therefore, expressions (4.4) and (4.5) were modified to obtain the half-power resolutions.

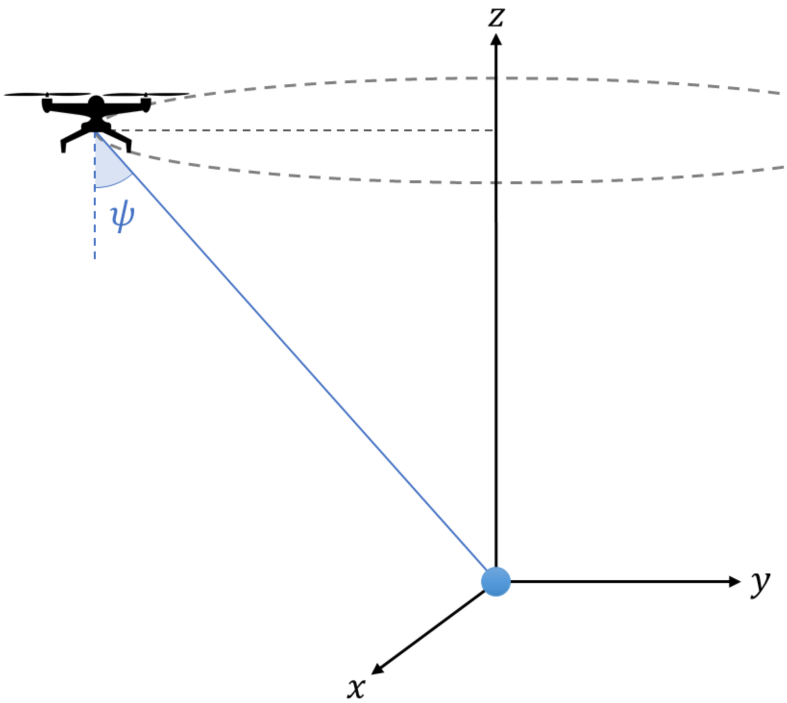


Figure 4.4 – Circular SAR: geometry for calculating ground and vertical resolutions.

4.1.4 Multi-Circular SAR

Although Circular SAR allows for 3D imaging, information is distorted by strong conical sidelobes perpendicular to the LOS direction. Therefore, further knowledge is required to build a focused 3D image from Circular SAR [44]. Multi-Circular SAR solves this issue with multiple circular flight tracks at different heights, thus providing additional information on elevation [26]. To put it another way, Multi-Circular SAR can be considered the combination of Circular SAR and SAR Tomography (see Figure 4.5). Indeed, this acquisition geometry is also known as Holographic SAR Tomography [26].

Multi-Circular SAR can provide full-3D reconstruction with very high resolution. The ground resolution is the same as a single circular flight path (4.4). As for the vertical resolution, (4.5) still applies, but the diversity in elevation angle stretches the effective bandwidth, thus improving the vertical resolution. Furthermore, SAR Tomography expressions (4.2) and (4.3) also apply for the resolution and maximum width perpendicular to the LOS direction. However, we need to replace the tomographic aperture (B) and the sampling distance

(ΔB) with their projections onto the direction perpendicular to the LOS [26]. Henceforth, these quantities shall be called the effective tomographic aperture ($B_\perp$) and the effective sampling distance ($\Delta B_\perp$).

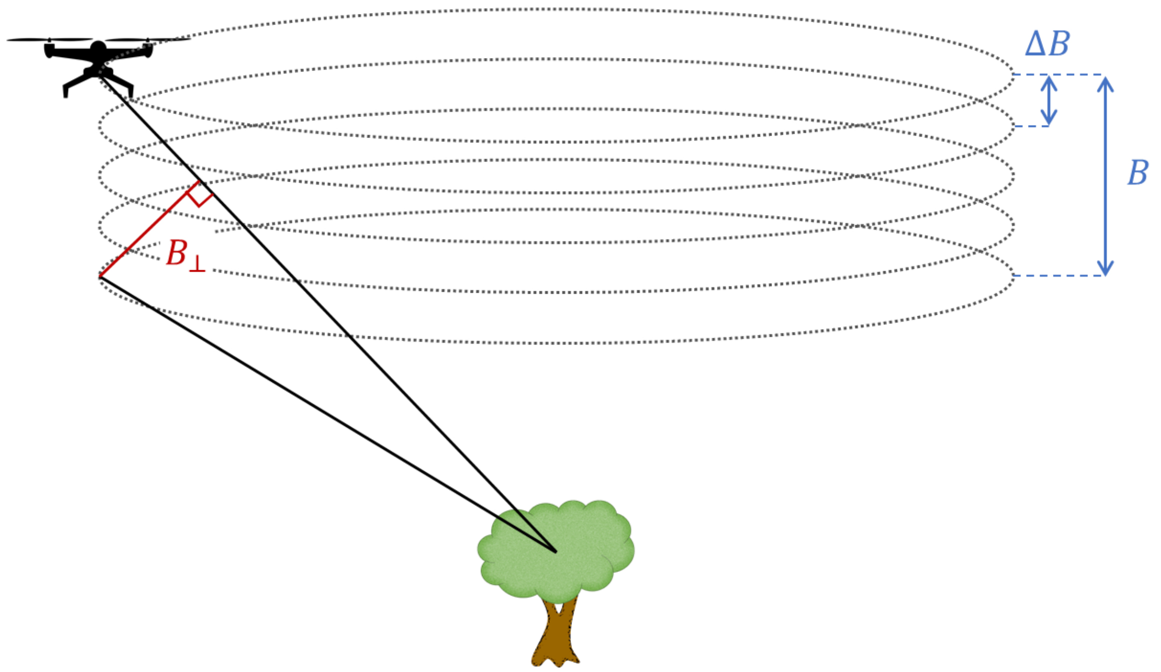


Figure 4.5 – Multi-Circular SAR: a combination of Circular SAR with SAR Tomography.

Multi-Circular SAR has been applied to studying forests with L-band data [26], crops with C- and X-band data [51], and glaciers with P- and L-band data [52]. The data were acquired with a SAR system mounted on an aircraft. In addition, the radii of the multiple circular flight paths were roughly constant.

4.1.5 Spiral SAR

The sampling distance hindered Multi-Circular SAR for a system working in the THz band. Fortunately, this issue was overcome with a cylindrical spiral flight path with constant vertical speed [53].

Furthermore, Sego *et al.* [54] studied different flight path geometries, including a cylindrical spiral, a planar spiral, and a spherical spiral. Nevertheless, their study did not evaluate the different parameters that may affect the performance of each flight path geometry. Later, in [55], those three geometries were considered for performing an experimental flight. However, only the cylindrical spiral was selected for practical reasons. Specifically, the cylindrical spiral path is easier to fly manually in the test aircraft, a 1952 Cessna 170.

To the best of the author’s knowledge, all multi-circular and spiral flight paths used in SAR surveys in literature had nearly constant radii. A probable reason is the limitations posed by aircraft flight mechanics. Nevertheless, drone flight mechanics are more flexible than aircraft, making spiral flight paths with variable radii possible. Suppose we combine the slanted acquisition geometry used in SAR Tomography with a spiral flight path. In that case, we will get a cone-shaped flight path whose radius increases as the height decreases. This configuration would have better vertical resolution than state-of-the-art solutions.

4.2 THE WAVENUMBER SHIFT

The wavenumber shift is a key concept for understanding the performance of Spiral SAR. The concept was first proposed for SAR Interferometry in [56]; then, it was applied to improve the ground resolution of linear flight paths in [57]. In both cases, the focus was the ground wavenumber shift. Later, in [26], the concept was also applied to Multi-Circular SAR, taking the vertical wavenumber shift into account to explain the improvement in vertical resolution. Unfortunately, although the reasoning was sound, the expressions provided in [26] are inaccurate. Therefore, this work proposes new, revised expressions.

Moreover, the wavenumber shift concept may clarify why Spiral SAR worked for a given system when Multi-Circular SAR could not [53], as pointed out in Subsection 4.1.5.

4.2.1 The Concept

This subsection shall explain the wavenumber shift concept for two interferometric antennas. Figure 4.6 illustrates the main idea, highlighting the different quantities at play.

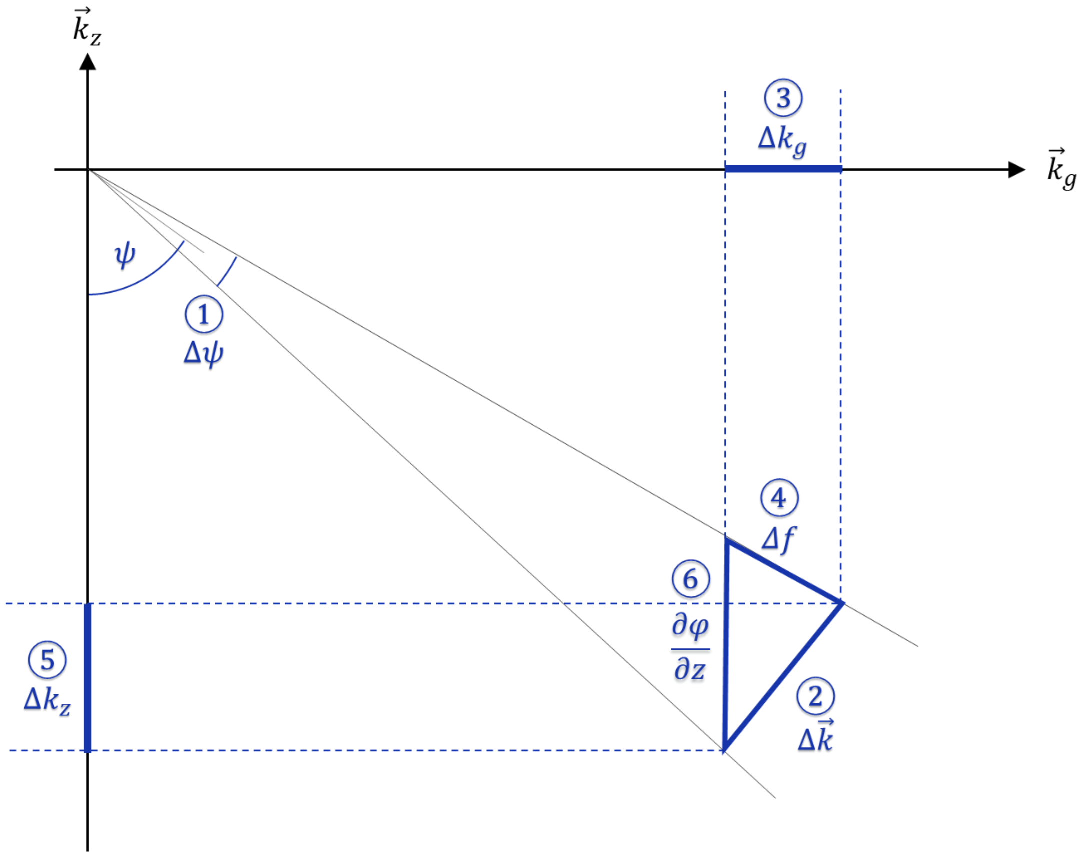


Figure 4.6 - The wavenumber shift and the quantities at play. Source: Modified from Gatelli *et al.* [56] © 1994 IEEE.

The quantities in Figure 4.6 shall be detailed in the following paragraphs. Nevertheless, here is a high-level summary:

- ①: A difference in look-angle ($\Delta\psi$) occurs when the radar sees the target from different locations (e.g., two interferometric antennas separated by a baseline).
- ②: The first leg of the triangle represents the wave vector shift ($\Delta\vec{k}$), i.e., the change in direction caused by the difference in look-angle.
- ③: The ground wavenumber shift (Δk_g) is the projection of the wave vector shift onto the ground plane.
- ④: The other leg of the triangle represents the equivalent wave vector that would produce the same ground wavenumber but would have been caused by a spectral shift (Δf).
- ⑤: The vertical wavenumber shift (Δk_z) is the projection of the wave vector shift onto the vertical direction.
- ⑥: The hypotenuse is the phase-to-height sensitivity ($\partial\varphi/\partial z$); it should not be confused with the vertical wavenumber shift.

4.2.1.1 The difference in look-angle

Two interferometric antennas look at the target from different angles due to the baseline separating them. As depicted in Figure 4.1, the difference in look-angle ($\Delta\psi$) can be approximated by:

$$\Delta\psi \approx \frac{b_\perp}{R_0} \tag{4.6}$$

where $b_\perp$ is the effective baseline, perpendicular to the LOS direction, and R_0 is the distance from the radar to the target. This approximation is valid for $b_\perp \ll R_0$.

4.2.1.2 The wave vector shift

The difference in look-angle makes the incident wave come from a different direction. In other words, it changes the direction of the wave vector. Indeed, the ground ($\vec{k}_g$) and vertical ($\vec{k}_z$) components of the wave vector are functions of the look-angle (ψ) [56]:

$$\left|\vec{k}_g\right| = k_g = 2k\sin\psi \tag{4.7}$$

$$\left|\vec{k}_z\right| = -k_z = 2k\cos\psi \tag{4.8}$$

where $k = 2\pi/\lambda$ is the signal wavenumber, and k_g and k_z are the ground and vertical wavenumbers, assuming a flat terrain.

4.2.1.3 The ground wavenumber shift

The difference in look-angle generates a shift in the ground wavenumber (Δk_g), which can be calculated using the first-order approximation below [56]:

$$\Delta k_g \approx \frac{4\pi}{\lambda}\cos\psi\,\Delta\psi \tag{4.9}$$

Substituting (4.6) into (4.9), we get:

$$\Delta k_g \approx \frac{4\pi b_\perp}{\lambda R_0}\cos\psi \tag{4.10}$$

4.2.1.4 The equivalent spectral shift

If the shift in ground wavenumber had been caused by a spectral shift (Δf) rather than a difference in look-angle, then we would have the following first-order approximation:

$$k_g = \frac{4\pi f}{c}\sin\psi \rightarrow \Delta k_g \approx \frac{4\pi}{c}\sin\psi\,\Delta f \tag{4.11}$$

Therefore, the equivalent spectral shift is [56]:

$$\Delta f = -\frac{c b_\perp}{\lambda R_0 \tan\psi} \tag{4.12}$$

In practice, if we compare the backscattered signal received by two interferometric antennas, the same components of the ground reflectivity spectrum will appear shifted by Δf. The spectral shift has two main effects: as Δf increases, the common part of spectra will decrease, but the overall bandwidth will increase. In (4.12), the negative sign represents the reduction in the common part of spectra. Moreover, (4.12) is identical to the interferogram fringe frequency [56].

If Δf becomes equal to the signal bandwidth (W), the two spectra will no longer overlap, the signals will become decorrelated, and interferometry will no longer be possible. In order to avoid this scenario, the baseline must not exceed the critical value [56]:

$$b_{\perp,c} = \frac{W\lambda}{c} R_0 \tan\psi \tag{4.13}$$

On the other hand, the spectral shift Δf can be exploited to improve the resolution of SAR images. For instance, for linear flight paths, the ground resolution (δ_g) is [57]:

$$\delta_g = \frac{c}{2W \sin\psi} \tag{4.14}$$

If the two spectra are combined, the bandwidth will increase by $|\Delta f|$, and the new ground resolution will be [57]:

$$\delta_g = \frac{c}{2W_g} \tag{4.15}$$

$$W_g = W \sin\psi + \Delta f_g \tag{4.16}$$

$$\Delta f_g = |\Delta f| \sin\psi = \frac{cb_\perp}{\lambda R_0} \cos\psi \tag{4.17}$$

where W_g and Δf_g are the effective bandwidth and the spectral shift in the ground plane. Note that (4.13) can also be found by making $\Delta f_g = W \sin\psi$.

4.2.1.5 The vertical wavenumber shift

Let us assume that the vertical wavenumber shift (Δk_z) is also produced by the difference in look-angle. Then, much like the ground wavenumber shift, it can be calculated with a first-order approximation:

$$\Delta k_z \approx \frac{4\pi}{\lambda} \sin\psi \, \Delta\psi \tag{4.18}$$

$$\Delta k_z \approx \frac{4\pi b_\perp}{\lambda R_0} \sin\psi \tag{4.19}$$

Equation (4.19) is a revised expression proposed in this work. This quantity is unrelated to SAR Interferometry but is essential to Multi-Circular SAR and Spiral SAR.

4.2.1.6 The phase-to-height sensitivity

The phase-to-height sensitivity is the interferometric phase change caused by a height change. It corresponds to the hypothenuse in Figure 4.6 and is given by [46], [56]:

$$\frac{\partial \varphi}{\partial z} = -\frac{4\pi\Delta f}{c\cos\psi} = \frac{4\pi\Delta\psi}{\lambda\sin\psi} \tag{4.20}$$

$$\frac{\partial \varphi}{\partial z} = \frac{4\pi b_{\perp}}{\lambda\, R_0 sin\,\psi} \tag{4.21}$$

Also, the phase-to-height sensitivity is associated with the height of ambiguity, which is the height change that causes a 2π change in the interferometric phase [46]:

$$z_{2\pi} = \frac{2\pi}{\partial\varphi/\partial z} = \frac{\lambda R_0}{2b_{\perp}}\sin\psi \tag{4.22}$$

Note that the phase-to-height sensitivity is not the same as the vertical wavenumber shift caused by the difference in look-angle (Δk_z). Rather, from Figure 4.6, it is equal to:

$$\frac{\partial \varphi}{\partial z} = \frac{\Delta k_g}{\tan\psi} + \Delta k_z \tag{4.23}$$

However, the phase-to-height sensitivity was identified in [56] as Δk_z. That choice probably led to expression (4.21) being used instead of (4.19) to determine the vertical resolution for Multi-Circular SAR [26].

4.2.2 The Wavenumber Shift in Multi-Circular and Spiral SAR

The wavenumber shift also occurs in Multi-Circular and Spiral SAR since the radar antenna looks at the target from different angles at each point across the tomographic aperture. However, a new distinction must be made, which did not apply to SAR Interferometry: the difference between the total and the incremental difference in look-angle. The first is due to the effective tomographic aperture ($B_{\perp}$) and the second is related to the effective sampling distance ($\Delta B_{\perp}$). Therefore, either $B_{\perp}$ or $\Delta B_{\perp}$ will replace the effective baseline ($b_{\perp}$) on the expressions of Subsection 4.2.1.

The sampling distance is either the gap between tracks in Multi-Circular SAR or the separation between turns in Spiral SAR. Suppose a spiral flight path has N_t turns, and a multi-circular flight path has $N_t + 1$ tracks. Then, if they have the same tomographic aperture (B), they will have the same sampling distance, $\Delta B = B/N_t$. Figure 4.7 shows an example for

$B = 40$ m and $N_t = 5$. Likewise, if the two flight paths have the same effective tomographic aperture ($B_\perp$), they will have the same effective sampling distance, $\Delta B_\perp = B_\perp / N_t$. Here, the word effective means perpendicular to the LOS direction.

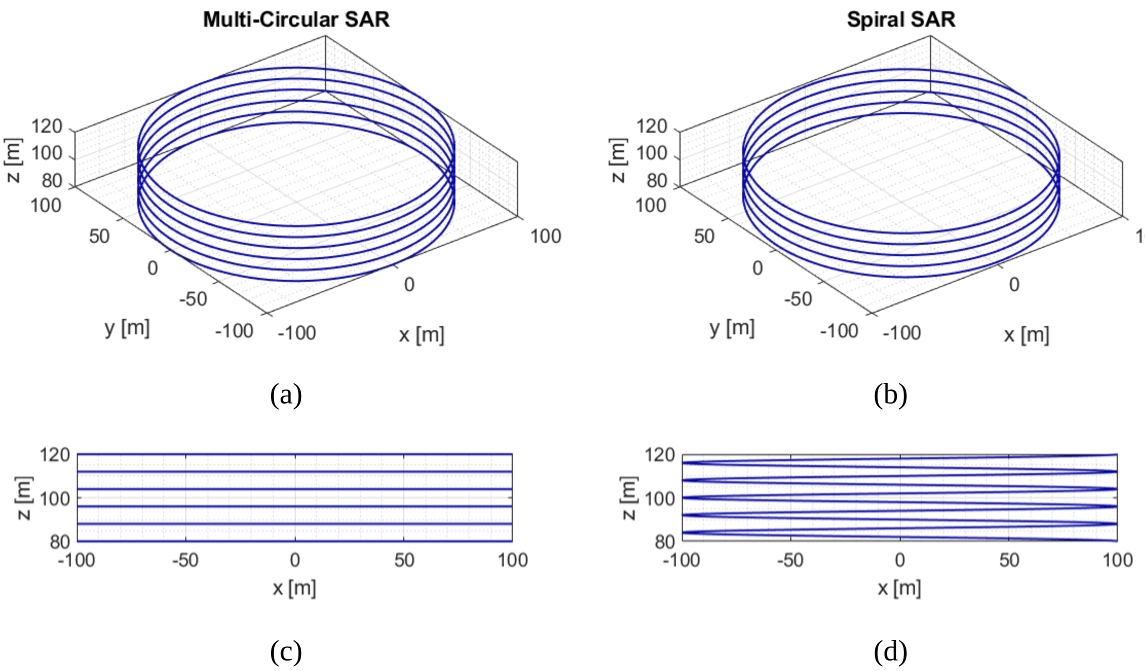


Figure 4.7 – Two flight paths with the same sampling distance: (a,b) perspective and (c,d) front views of (a,c) a multi-circular flight path with six tracks and (b,d) a spiral flight path with five turns.

4.2.2.1 The vertical resolution

Multi-Circular SAR and Spiral SAR can achieve better vertical resolutions than Circular SAR, thanks to the entire tomographic aperture widening the vertical bandwidth [26]. With this in mind, in order to find an expression for the vertical resolution, first, we need a new expression for the vertical wavenumber shift:

$$\Delta k_z(B_\perp) \approx \frac{4\pi B_\perp}{\lambda R_0} \sin\psi \tag{4.24}$$

Then, from Δk_z, we can derive the effective bandwidth (W_z) and the spectral shift (Δf_z) in the vertical direction:

$$W_z = W \cos\psi + \Delta f_z \tag{4.25}$$

$$\Delta f_z(B_\perp) = \frac{c B_\perp}{\lambda R_0} \sin\psi \tag{4.26}$$

Note that (4.25) and (4.26) are similar to (4.16) and (4.17), swapping sines and cosines. Finally, the vertical resolution is given by:

$$\delta_z = \sqrt{\frac{\ln(2)}{\pi}} \frac{c}{W_z} \tag{4.27}$$

Figure 4.8 illustrates the spectra for two different look-angles in the frequency domain. It highlights the effective bandwidths and spectral shifts in the ground plane and the vertical direction.

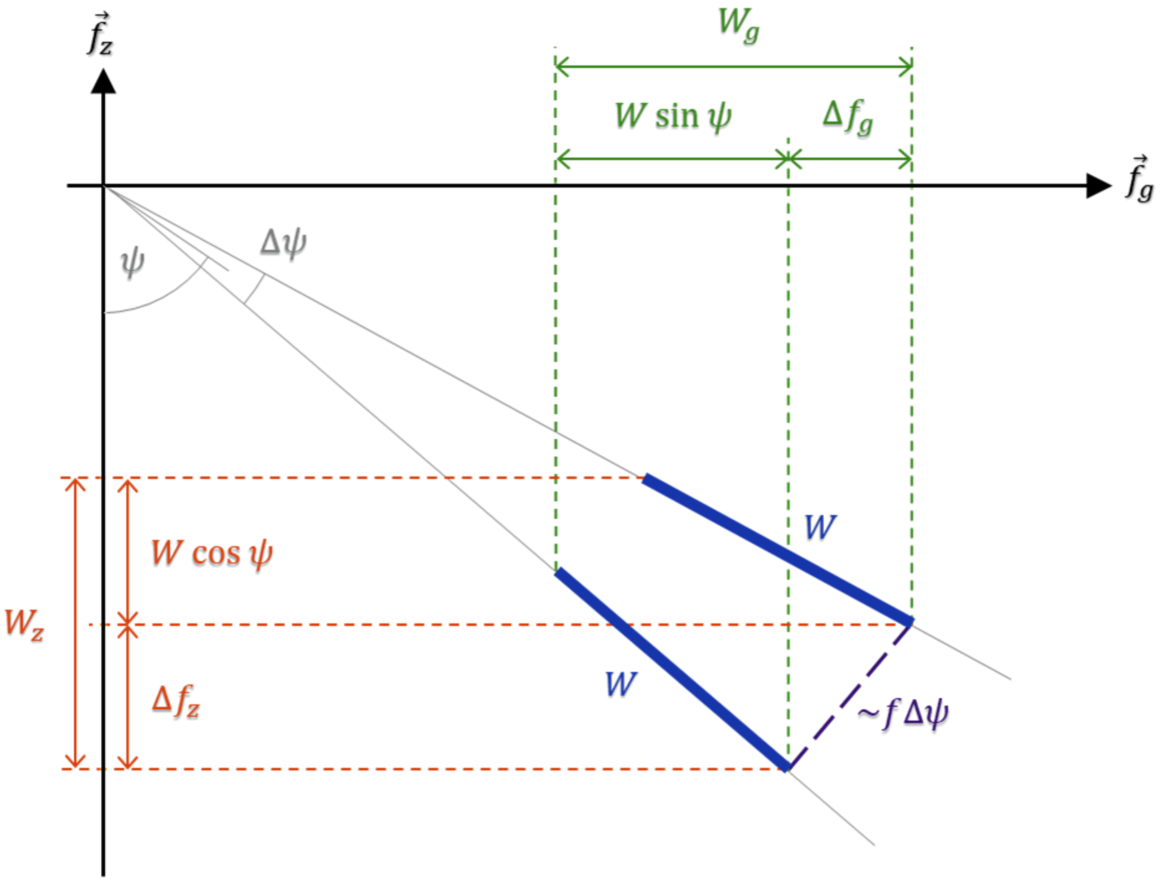


Figure 4.8 – The effective bandwidths and spectral shifts in the ground plane and the vertical direction.

4.2.2.2 The critical sampling distance

The critical sampling distance may help explain why Spiral SAR worked in conditions where Multi-Circular SAR did not [53]. In SAR Interferometry, the critical baseline is the minimum baseline for which the ground spectra do not overlap. In Multi-Circular SAR, the critical sampling distance also considers the vertical spectra [26].

Suppose that instead of representing the beginning and end of the tomographic aperture, Figure 4.8 represents two consecutive tracks in Multi-Circular SAR. Then, if either one of the following expressions is true, the tracks will become decorrelated:

$$\Delta f_g(\Delta B_\perp) \geq W \sin\psi \tag{4.28}$$

$$\Delta f_z(\Delta B_\perp) \geq W\, cos\, \psi \tag{4.29}$$

To avoid this scenario, the effective distance between tracks should not surpass the critical value:

$$\Delta B_{\perp,c} = \min\left(\frac{W\lambda}{c} R_0 \tan\psi\,, \quad \frac{W\lambda}{c}\frac{R_0}{\tan\psi}\right) \tag{4.30}$$

$$\Delta B_{\perp,c} = \begin{cases} \dfrac{W\lambda}{c} R_0 \tan\psi\,, & 0 \leq \psi < 45° \\ \dfrac{W\lambda}{c}\dfrac{R_0}{\tan\psi}, & 45° \leq \psi < 90° \end{cases} \tag{4.31}$$

Note that $W\lambda/c$ is the fractional bandwidth, whose values are typically low for SAR systems. Furthermore, the distance R_0 will be small for most drone operations, which are usually not allowed to fly higher than 120 m (400 feet) above ground level [58]–[60].

Therefore, Multi-Circular SAR acquisitions require multiple tracks to keep the sampling distance below the critical value for drone operations. However, flying multiple tracks could potentially come across another issue: the drone's limited autonomy. Indeed, the drone-borne SAR system of Section 1.1 has an autonomy of about 20 min.

Fortunately, decorrelation might not affect Spiral SAR because the wavenumber shift is almost continuous for isotropic targets. After all, the sampling would not be related to the distance between spiral turns but to the pulse repetition frequency. Figure 4.9 shows the wavenumber domain for a multi-circular flight path with five tracks and a spiral flight path with four turns. For both flight paths, $B_\perp = 50$ m, $\Delta B_\perp = 12.5$ m, $R_0 = 165$ m, and $\psi = 55°$. Also, the fractional bandwidth is 4.7 % in both cases. Therefore, $\Delta B_\perp$ is above the critical value ($\Delta B_{\perp,c} = 5.4$ m).

While Figure 4.9(e) shows clear gaps in k_z, in Figure 4.9(f), k_z varies continuously throughout the flight path. Furthermore, because $\psi > 45°$, $\Delta B_{\perp,c}$ is defined by Δf_z, not Δf_g. This dependence on Δf_z can be verified when comparing Figure 4.9(c) and Figure 4.9(e).

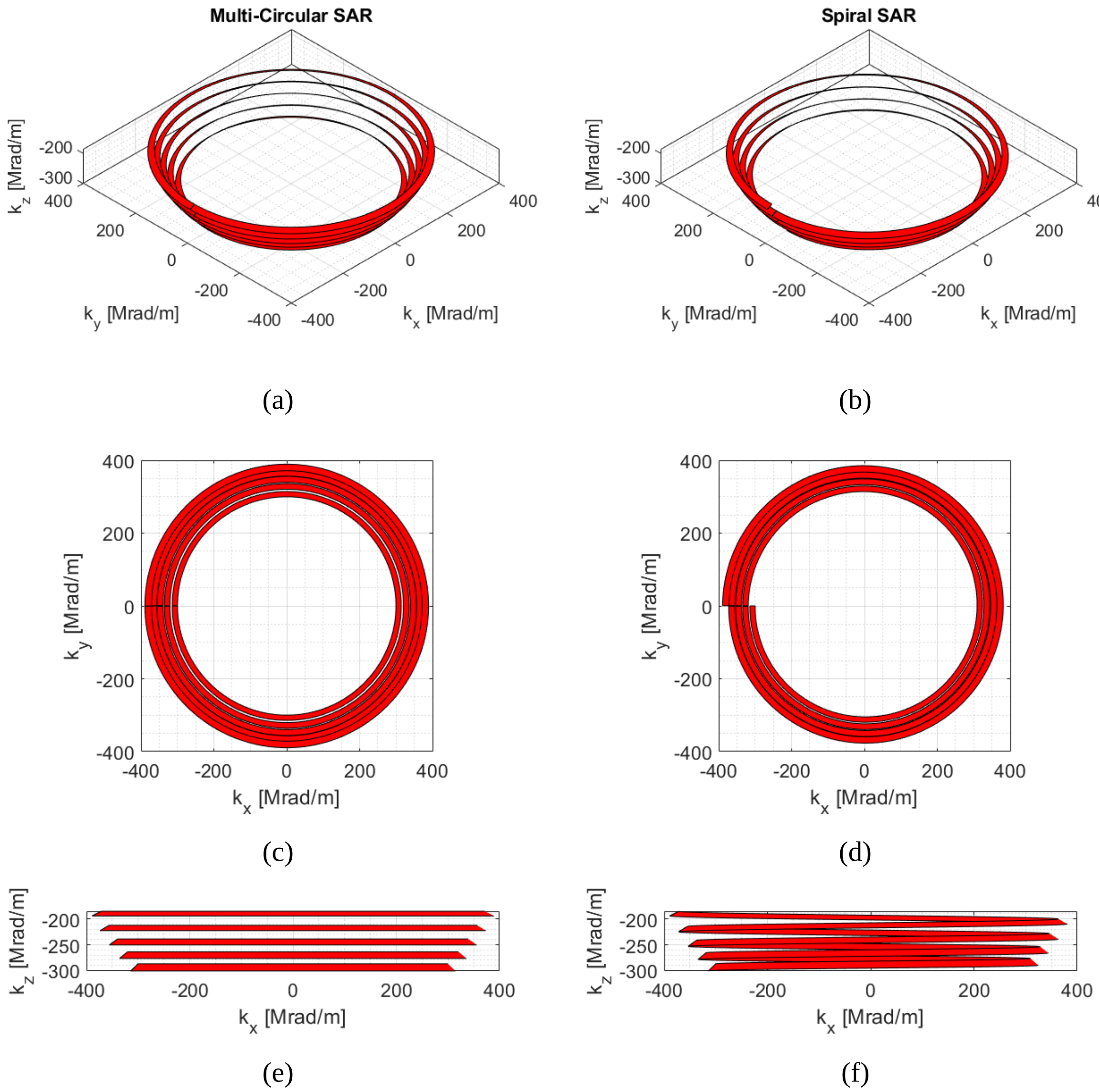


Figure 4.9 – Wavenumber domain for (a,c,e) a Multi-Circular SAR acquisition and (b,d,f) a Spiral SAR acquisition: (a,b) perspective, (c,d) top, and (e,f) front views.

4.2.2.3 The height of ambiguity

Although Spiral SAR might not be affected by decorrelation, $\Delta B_{\perp}$ still contributes to the height of ambiguity:

$$z_{2\pi} = \frac{\lambda R_0}{2\Delta B_{\perp}} \sin\psi \tag{4.32}$$

$$z_{2\pi} = \frac{N_t \lambda R_0}{2B_{\perp}} sin\,\psi \tag{4.33}$$

Note that (4.32) is SAR Tomography's maximum width perpendicular to the LOS direction (4.3) projected onto the *z*-direction.

4.3 Conical Spirals and Their Parameters

Section 4.1 argued that conical flight paths would perform better than cylindrical flight paths, i.e., those with constant radii. Furthermore, Section 4.2 hypothesized that decorrelation might not be an issue for Spiral SAR when working with isotropic targets. Moreover, Section 4.2 proposed a revised expression for the vertical resolution of Multi-Circular SAR and Spiral SAR. That expression is a function of the effective tomographic aperture, which needs to be further detailed.

In order to address all these matters, this section uses several simulation results. All simulation scenarios were processed with the FFBP algorithm presented in Section 3.2, written in MATLAB R2018a, and executed on an Intel(R) Core(TM) i7-7700 CPU (3.60 GHz) with 64 GB RAM.

4.3.1 Spiral Coordinates

Before showing the simulation results, let us determine the coordinates of a spiral flight path from its geometric parameters. Figure 4.10 shows an illustrative example of a conical spiral flight path. The spiral moves downwards from the height at the top (z_{top}) to the height at the base (z_{base}), making four complete turns ($N_t = 4$).

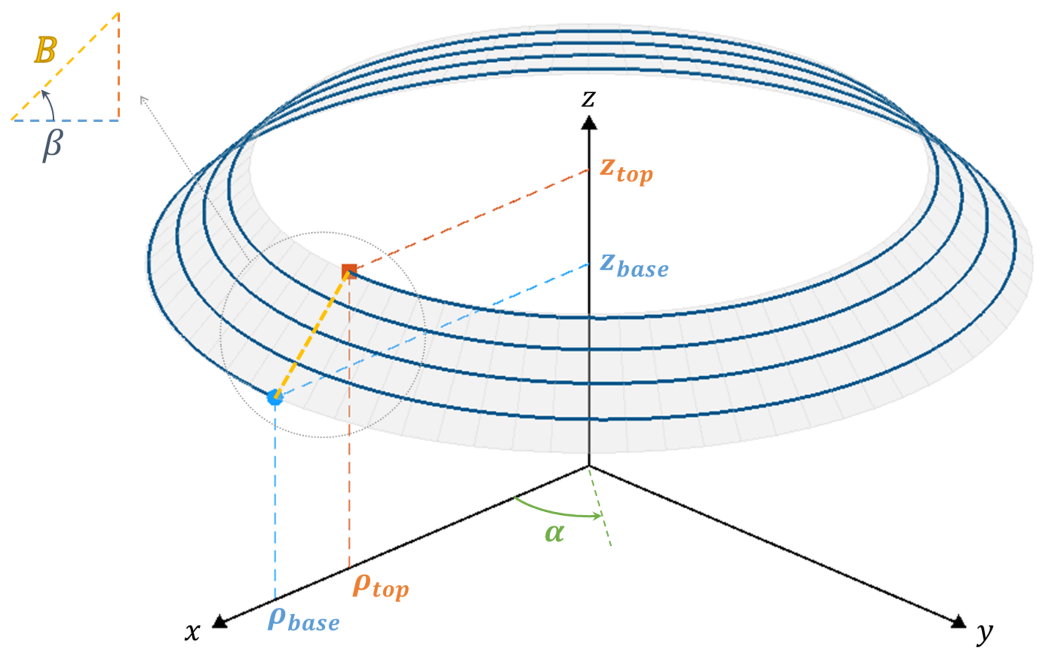


Figure 4.10 – A conical spiral flight path and its parameters: the top and base height (z_{top} and z_{base}) and radii (ρ_{top} and ρ_{base}), the azimuth angle (α), the tomographic aperture (B) and the tilt angle (β).

Consider that the spiral radius varies at a constant rate from the top (ρ_{top}) to the base (ρ_{base}). In that case:

$$\rho(t) = \rho_{top} + V_\rho t, \quad 0 \le t < t_{max} \tag{4.34}$$

$$V_\rho = \frac{\rho_{base} - \rho_{top}}{t_{max}} \tag{4.35}$$

where $\rho(t)$ is the spiral radius as a function of time, V_ρ is the radial speed, and t_{max} is the time it takes to complete the spiral. Note that, for this example, $\rho_{base} > \rho_{top}$, so $V_\rho > 0$.

Now, suppose that a drone flies through the conical spiral path with a constant tangential speed (V_0). Then, we can determine the angular speed and, thus, the azimuth angle:

$$\frac{\partial\alpha(t)}{\partial t} = \frac{V_0}{\rho(t)} \quad \rightarrow \quad \alpha(t) = \int_0^t \frac{V_0}{\rho(\tau)} d\tau \tag{4.36}$$

where $\alpha(t)$ is the azimuth angle as a function of time. From (4.34) and (4.36), we get:

$$\alpha(t) = \frac{V_0}{V_\rho}\left[ln\big(\rho(t)\big) - ln\big(\rho_{top}\big)\right], \quad 0 \le \alpha(t) \le 2\pi N_t \tag{4.37}$$

Note that $\alpha(t_{max}) = 2\pi N_t$ and that $\rho(t_{max}) = \rho_{base}$. Thus, substituting (4.35) into (4.37) for $t = t_{max}$, we obtain:

$$t_{max} = \frac{2\pi N_t}{V_0} \frac{\rho_{base} - \rho_{top}}{ln(\rho_{base}) - ln\big(\rho_{top}\big)} \tag{4.38}$$

In the case of a cylindrical spiral, the radius is constant, i.e., $\rho_{base} = \rho_{top} = \rho_0$. Therefore, expressions (4.37) and (4.38) become much more straightforward:

$$\alpha(t) = \frac{V_0}{\rho_0} t \tag{4.39}$$

$$t_{max} = \frac{2\pi N_t}{V_0} \rho_0 \tag{4.40}$$

Now that we have determined the radius and the azimuth angle of the spiral flight path, we need an expression for its height. If we consider that the height also varies at a constant rate, then we get:

$$z(t) = z_{top} + V_z t \tag{4.41}$$

$$V_z = -\frac{(z_{top} - z_{base})}{t_{max}} \tag{4.42}$$

where $z(t)$ is the spiral height as a function of time, and V_z is the vertical speed. Note that, because the flight path moves downward, $V_z < 0$.

Next, we can obtain the cartesian coordinates of the spiral flight path using the following expressions:

$$x(t) = \rho(t)\cos(\alpha(t)) \tag{4.43}$$

$$y(t) = \rho(t)\sin(\alpha(t)) \tag{4.44}$$

From a design perspective, it might be more interesting to describe the conical spiral flight path using other parameters, such as the mean height (z_0), the mean radius (ρ_0):

$$z_0 = \frac{z_{base} + z_{top}}{2} \tag{4.45}$$

$$\rho_0 = \frac{\rho_{base} + \rho_{top}}{2} \tag{4.46}$$

Other important parameters are the tomographic aperture (B) and the tilt angle (β), both depicted in Figure 4.10 as elements of a right triangle:

$$B = \sqrt{(z_{top} - z_{base})^2 + (\rho_{top} - \rho_{base})^2} \tag{4.47}$$

$$\beta = \tan^{-1}\left(\frac{z_{top} - z_{base}}{\rho_{base} - \rho_{top}}\right) \tag{4.48}$$

Finally, instead of z_0 and ρ_0, we can describe the conical spiral flight path using the corresponding look-angle (ψ_0) and the mean distance (R_0) from the radar to a target at the origin:

$$\psi_0 = \tan^{-1}\left(\frac{\rho_0}{z_0}\right) \tag{4.49}$$

$$R_0 = \sqrt{{z_0}^2 + {\rho_0}^2} \tag{4.50}$$

4.3.2 The Effective Tomographic Aperture

In the last subsection, we have established that the tomographic aperture (B) is a geometric parameter of the conical spiral flight path. However, B does not directly affect the wavenumber shift. From Section 4.2, the wavenumber shift, and thus the vertical resolution, are functions of the effective tomographic aperture ($B_\perp$) instead.

We know that $B_\perp$ should be perpendicular to the LOS direction from an analogy with interferometry. The only issue here is how to define the LOS direction. This definition will affect how we measure the distance between the radar and the target, thus impacting how we calculate the vertical resolution. Unfortunately, we cannot assume that the distance variation is negligible or that $B_\perp \ll R_0$. First, the tomographic aperture is larger than an interferometric baseline. Second, when working with drones, the distances from the radar to the imaged area are way shorter than for aircraft or satellites.

Figure 4.11 shows the proposed solution. First, we define the LOS direction from the midpoint of the tomographic aperture to a target at the origin. The magnitude of this vector is the mean distance (R_0) from the spiral flight path to the origin. Then, we project B onto the direction perpendicular to the LOS:

$$B_\perp = B|\cos(\beta - \psi_0)| \tag{4.51}$$

where β is the tilt angle, and ψ_0 is the midpoint look-angle. The reason for taking the absolute value is that $B_\perp$ will be positive even when the cosine is negative ($|\beta - \psi_0| > 90°$).

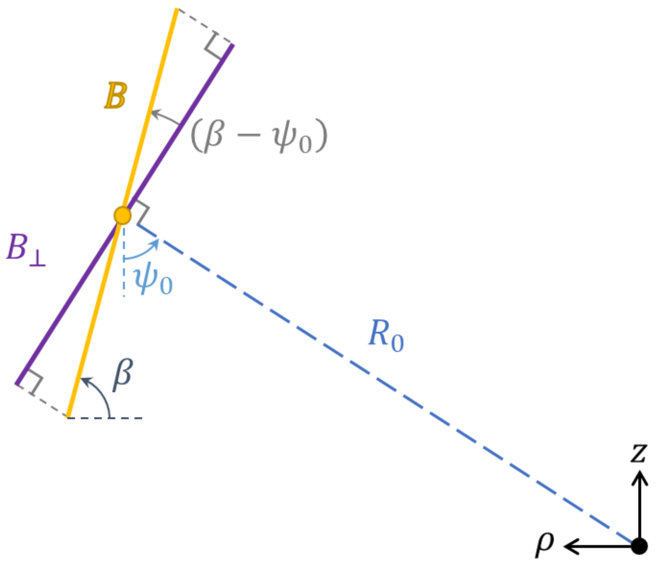


Figure 4.11 – The effective tomographic aperture ($B_\perp$) as determined from the midpoint look-angle (ψ_0), B and β. $B_\perp$ is perpendicular to the LOS direction, which is defined using the mean distance (R_0).

4.3.2.1 Simulation setup

Table 4.1 shows the setup parameters for the simulations presented in this subsection. The simulation batch consisted of 19 spiral flight paths with the same effective tomographic aperture ($B_\perp$), mean height (z_0) and mean radius (ρ_0). Thus, the midpoint look-angle (ψ_0) and the mean distance (R_0) were kept constant. On the other hand, the tilt angle (β) varied from 0° to 90° in increments of 5°; thus, the angle $\beta - \psi_0$ also varied. Therefore, to keep $B_\perp$ constant, the tomographic aperture (B) had to be adjusted according to (4.51). The number of turns was also kept constant. In addition, it should be noted that the overall speed replaced the tangential speed (V_0) in equations (4.36-4.40) because the radial and vertical components of speed are much smaller than the tangential component.

Table 4.1 - Parameters of the simulation batch with constant $B_\perp$.

	Parameter		Values	Units
Flight paths	$B_\perp$	Effective tomographic aperture	50	m
	z_0	Mean height	95	m
	ρ_0	Mean radius	135	m
	β	Tilt angle	[0, 90]	degrees
	$\Delta\beta$	Tilt angle increment	5	degrees
	N_t	Number of turns	10	-
	V_0	Drone speed	7.5	m/s
Radar	λ	Wavelength	70.54	cm
	W	Bandwidth after range compression	20	MHz
	Δt	Pulse repetition interval	6.6	ms
	Δr	Range sampling interval	2.456	m
FFBP setup		Number of voxels at the output image	81 × 81 × 36	-
		Output image resolution	5 × 5 × 30	cm³
	$(x_t, 0, 0)$	Output image center	(0, 0, 0)	m
			(20, 0, 0)	m
			(40, 0, 0)	m
	L	Subapertures combined at each recursion	3	-
		First split into image blocks	1 × 1 × 1	-

Figure 4.12 shows two flight paths from the simulation batch. The first (blue) is a conical spiral for which $\beta = \psi_0$, thus $B = B_\perp$. The second (yellow) represents the state-of-the-

art: a cylindrical spiral. Figure 4.12(c) indicates that B is about 60 m for the cylindrical spiral flight path, 20% longer than $B_{\perp}$. Such a long tomographic aperture would become an issue in real life since the flight path would not comply with the flight ceiling requirement, which must be kept below 120 m.

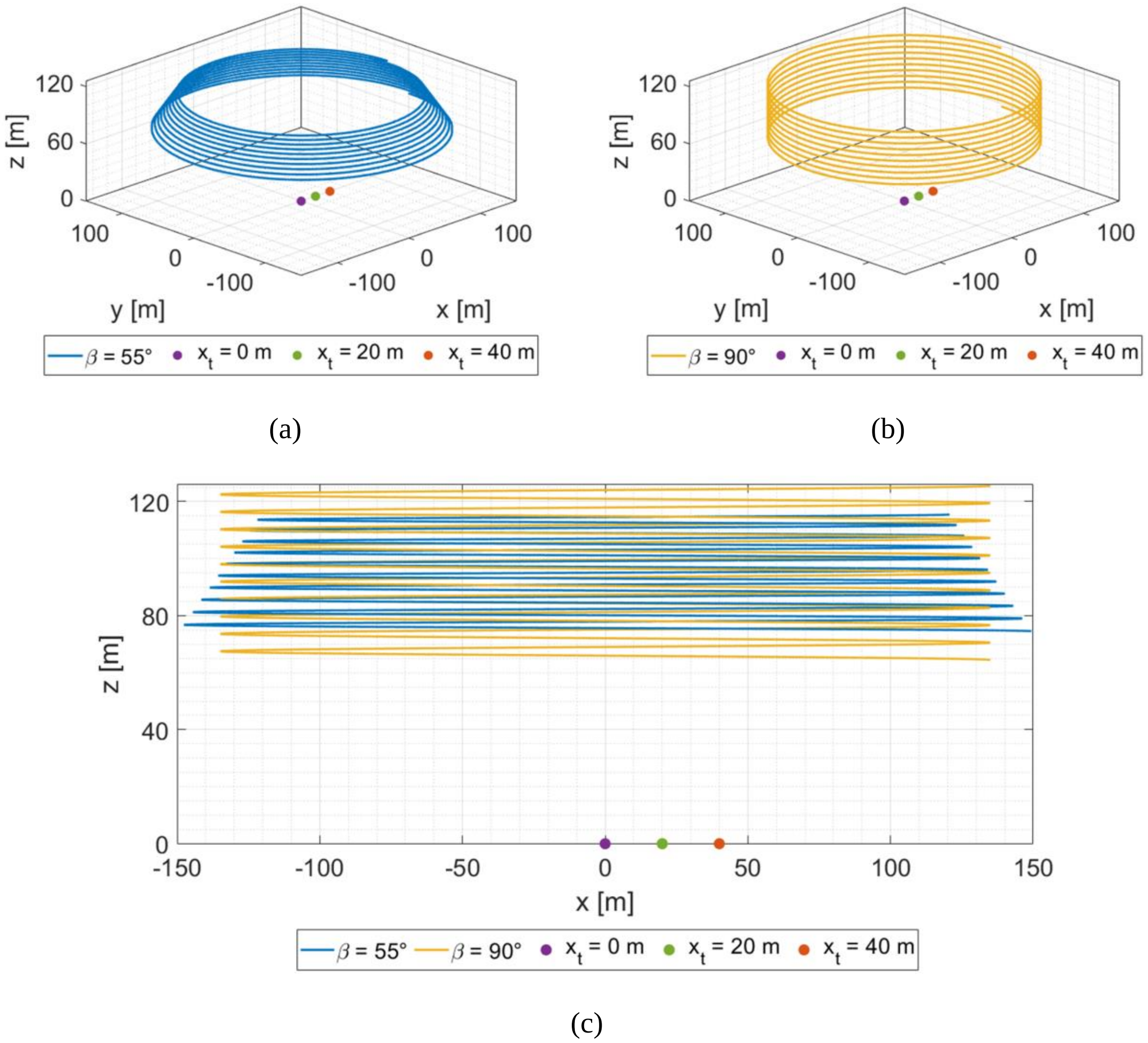


Figure 4.12 – Spiral flight paths with the same $B_{\perp}$: (a) the conical spiral with $B = B_{\perp}$ ($\beta = 55°$); (b) the cylindrical spiral ($\beta = 90°$); (c) a comparison of the two flight paths. There are three isotropic targets in the scene, at $x_t = \{0, 20, 40\}$ m.

The simulation batch considered a drone operating in P-band, using parameters from a real-life drone-borne SAR system (see Table 3.6). It is worth noting that, although the chirp bandwidth is 50 MHz, the bandwidth after range compression (W) is 40% of this value (20 MHz) due to the Hamming window (see Subsection 2.2.2). The simulations also took into account the two-way path loss of radar systems (2.6).

Furthermore, the simulation batch considered three isotropic targets in the scene, each placed on the positive x-axis, at $x_t = \{0, 20, 40\}$ m, as shown in Figure 4.12. There was only one raw radar data for each flight path, but the volumes around each target were processed separately. Each output image is 4.05 × 4.05 × 10.8 m³. Due to their small volume, splitting them into image blocks was unnecessary. Finally, the FFBP was set up with $L = 3$ subapertures combined at each recursion.

4.3.2.2 Simulation results

Figure 4.13 shows the resolutions in space for each target against the tilt angle (β). Note that all the resolution curves are nearly constant, whether ground (δ_x and δ_y) or vertical (δ_z). Recall that the vertical resolution depends on $B_\perp$, W, λ, R_0 and ψ_0 (4.25-4.27). Because all these values were kept constant, a constant vertical resolution was expected, as indicated by the theoretical value curve in Figure 4.13(c).

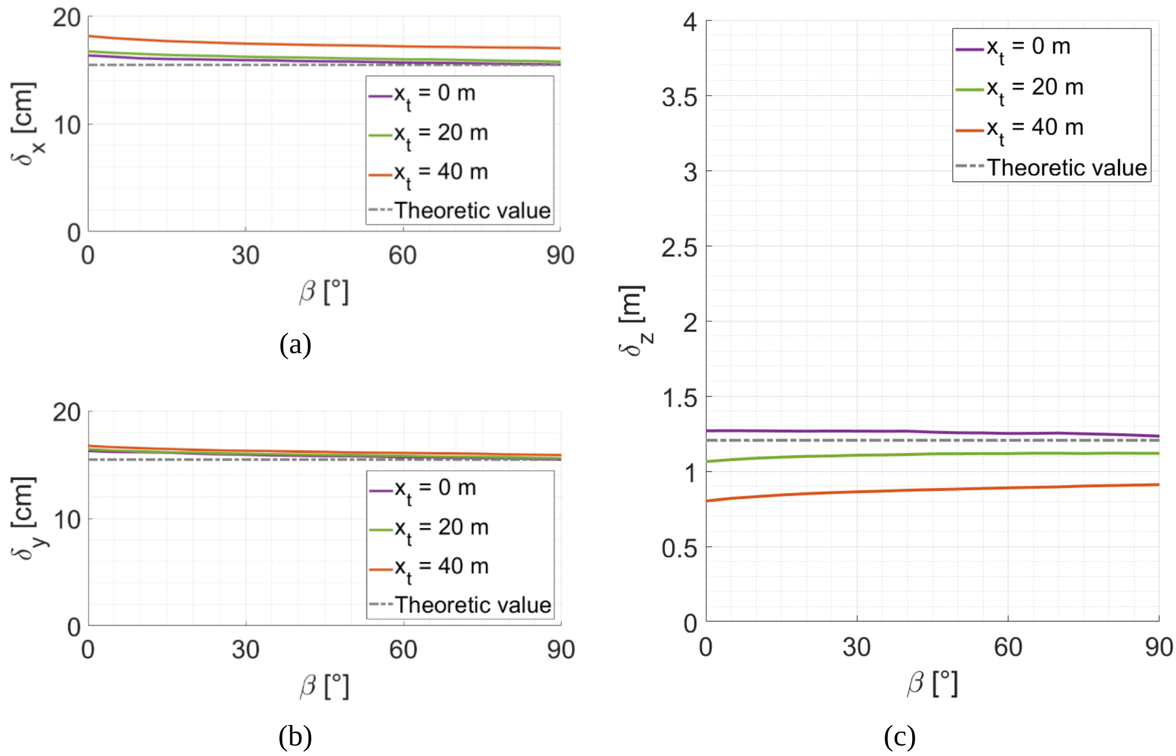


Figure 4.13 – Resolutions in space against β for the simulation batch with constant $B_\perp$: ground resolutions, (a) δ_x and (b) δ_y; (c) vertical resolution (δ_z).

Because the ground resolution does not depend on the bandwidth, it is not affected by the wavenumber shift. Therefore, the theoretical value curves in Figure 4.13(a,b) correspond to the ground resolution of a singular flight path, determined by the mean radius (ρ_0) and the mean height (z_0). Recall that the ground resolution depends on λ and ψ_0 (4.4). Once again, the expected result was a constant ground resolution because these values were kept constant.

In Figure 4.13(b), δ_y is almost identical for all three targets. On the other hand, in Figure 4.13(a,c), as the target moves away from the center, δ_x worsens, but δ_z improves. This behavior suggests that the radar positions closer to the target have a greater impact on the resolution. To make this statement clearer, consider the equivalent look-angle ($\tilde{\psi}_0$) and the equivalent bandwidth ($\widetilde{W}$) below:

$$\tilde{\psi}_0 = \tan^{-1}\left(\frac{\rho_0 - |x_t|}{z_0}\right) \tag{4.52}$$

$$\widetilde{W} = \frac{W_z}{\cos(\psi_0)} \tag{4.53}$$

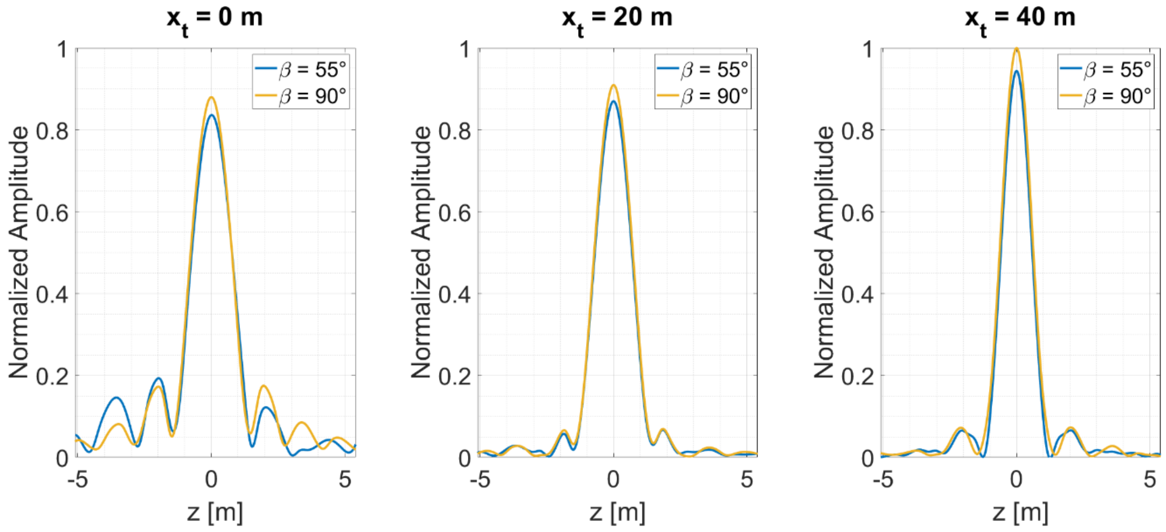
where W_z is the effective bandwidth in the vertical direction (4.25). Suppose we calculate the theoretic values of δ_x and δ_z considering a single circle flight path, with look-angle $\tilde{\psi}_0$ and bandwidth $\widetilde{W}$ (4.4,4.5). In that case, the RMSE will improve, as shown in Table 4.2. Note that for $x_t = 0$ m, $\tilde{\psi}_0 = \psi_0$.

Table 4.2 – RMSE for the simulation batch with constant $B_\perp$, with and without considering the near-range influence.

Target	**Theoretical values with $\tilde{\psi}_0$**		**RMSE with ψ_0**			**RMSE with $\tilde{\psi}_0$**	
	δ_x [cm]	**δ_z [m]**	**x [cm]**	**y [cm]**	**z [m]**	**x [cm]**	**z [m]**
$x_t = 0$ m	15.46	1.21	0.36	0.36	0.05	0.36	0.05
$x_t = 20$ m	16.40	1.09	0.68	0.46	0.10	0.26	0.02
$x_t = 40$ m	17.88	0.98	1.92	0.75	0.33	0.50	0.11

At first glance, path loss seems to be behind the behavior described above. However, it also occurs when path loss is not taken into account. Therefore, the explanation may lie elsewhere: the phase contributions from near-range and far-range are different because the sensitivity to a change in target position is not the same.

On the other hand, path loss affects the amplitude of the output image. Figure 4.14 shows the normalized amplitude at the z-axis, comparing the response of each target obtained with the two flight paths in Figure 4.12. All curves in Figure 4.14 were normalized to the same maximum value, corresponding to the outermost target and the cylindrical flight path. In addition, these curves were upsampled with a factor of 10.



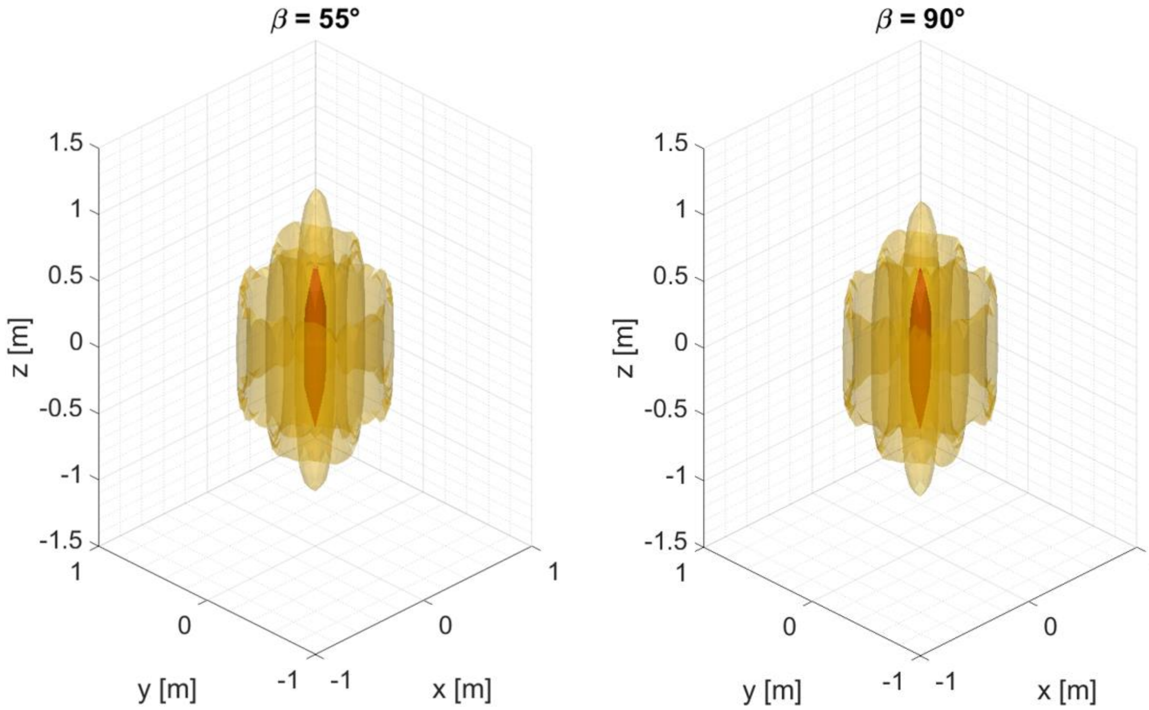
(a) (b) (c)

Figure 4.14 – Normalized amplitude at the z-axis for two flight paths of the simulation batch with constant $B_{\perp}$: the conical spiral with $B = B_{\perp}$ ($\beta = 55°$) and the cylindrical spiral ($\beta = 90°$). Each panel corresponds to the response of a different target: (a) $x_t = 0$ m; (b) $x_t = 20$ m; (c) $x_t = 40$ m.

Note that the amplitude increases as the target moves away from the center. Also, note that the sidelobes are stronger for the target at $x_t = 0$ m. The most probable reason is that this target is right at the spiral flight path axis. Close to the center, the distance variation between the radar and the target is smaller, leading to the constructive sum of different contributions. Furthermore, the two flight paths produce similar results, even though the response is slightly stronger for the cylindrical flight path.

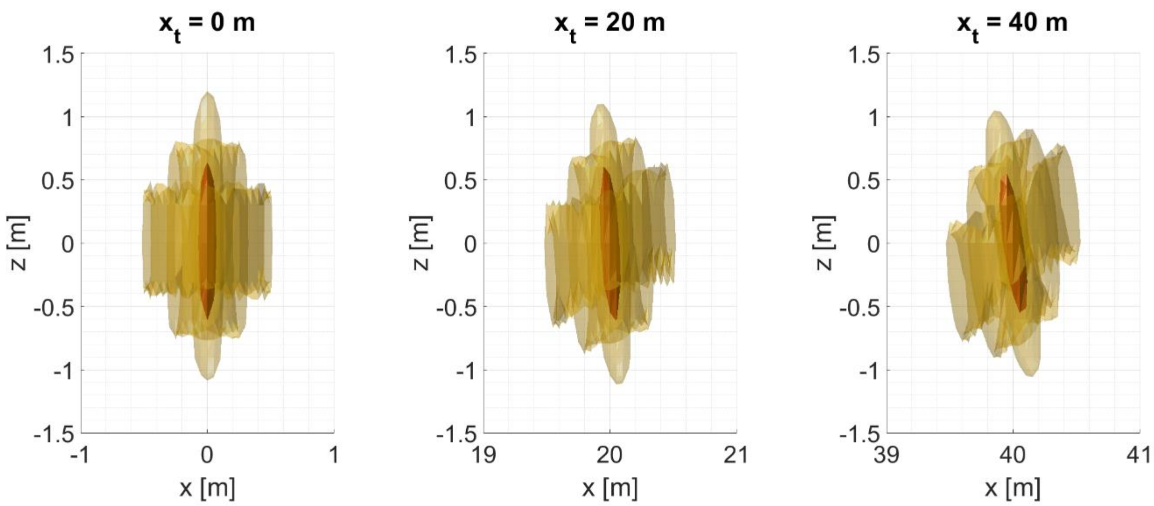
Figure 4.15 shows the -3 dB (red) and the -13 dB (yellow) isosurfaces for the target at the center, comparing the responses of the two flight paths in Figure 4.12. The values in dB were calculated using the maximum amplitude of each volume as a reference. Note how similar the two responses are. Again, this similarity is expected because both cases have the same $B_{\perp}$, W, λ, R_0 and ψ_0.

Figure 4.16 shows the same isosurfaces for the flight path with $\beta = 55°$, comparing the responses of each target. Note that the response is upright for the target at $x_t = 0$ m and becomes increasingly tilted as the target moves away from the center. The sidelobes tend to spread in the LOS direction, and the main lobe approaches the direction perpendicular to the LOS. Also, note that the centers of the volumetric images always appear at the correct positions despite this tilting. This result is yet another evidence of the radar positions at near-range exerting more influence over the output image.



(a) (b)

Figure 4.15 – Output images for the target at $x_t = 0$ m, obtained with two flight paths from the simulation batch with constant $B_\perp$: (a) the conical spiral with $B = B_\perp$ ($\beta = 55°$); (b) the cylindrical spiral ($\beta = 90°$). The -3 dB isosurfaces are opaque red, and the -13 dB isosurfaces are translucent yellow.



(a) (b) (c)

Figure 4.16 – Output images obtained with the conical spiral flight path for which $B = B_\perp$ ($\beta = 55°$). The -3 dB isosurfaces are opaque red, and the -13 dB isosurfaces are translucent yellow. Each panel corresponds to the response of a different target: (a) $x_t = 0$ m; (b) $x_t = 20$ m; (c) $x_t = 40$ m.

4.3.3 The Effect of Tilt Angle

The last subsection showed that conical and cylindrical flight paths could produce similar results with the same effective tomographic aperture ($B_{\perp}$). However, the tomographic aperture (B) of the cylindrical flight path was about 20% longer than the conical flight path with $B = B_{\perp}$. This subsection will explore what happens when we keep B constant and vary the tilt angle (β) to demonstrate that conical flight paths perform better than cylindrical flight paths under those conditions.

4.3.3.1 Simulation setup

Table 4.3 shows the setup parameters that changed from those in Table 4.1. The simulation batch consisted of 36 spiral flight paths with the same three isotropic targets as before, located at $x_t = \{0, 20, 40\}$ m. Figure 4.17 compares the conical spiral flight path for which $B = B_{\perp}$ (blue), and the cylindrical spiral flight path (yellow).

Table 4.3 – Parameters of the simulation batch with constant B.

	Parameter	Values	Units
B	Tomographic aperture length	50	m
β	Tilt angle	[0, 175]	degrees
$\Delta\beta$	Tilt angle increment	5	degrees

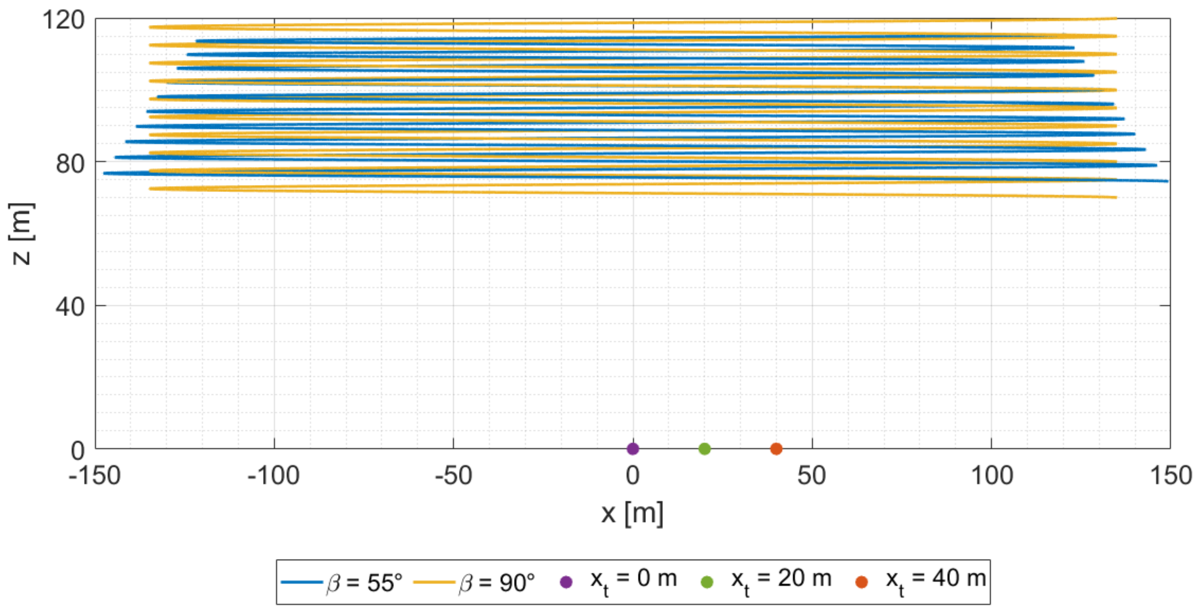
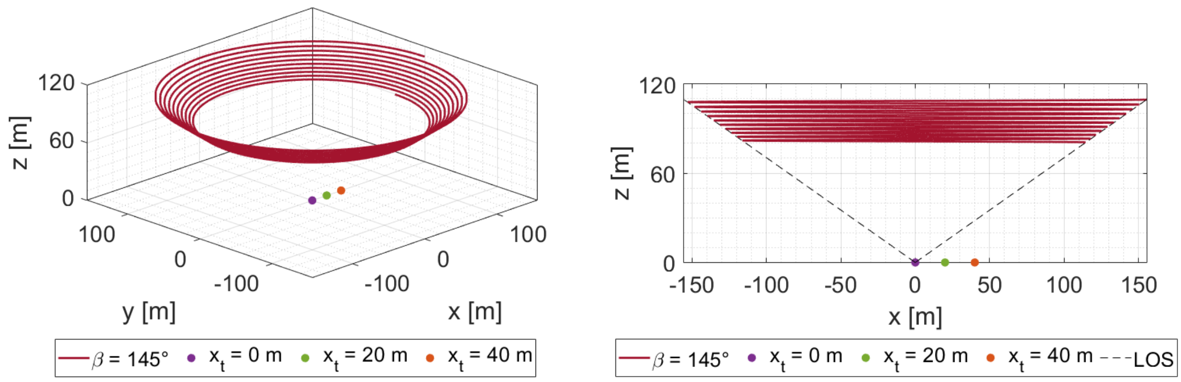


Figure 4.17 – Spiral flight paths with the same B: the conical spiral with $B = B_{\perp}$ ($\beta = 55°$) and the cylindrical spiral ($\beta = 90°$).

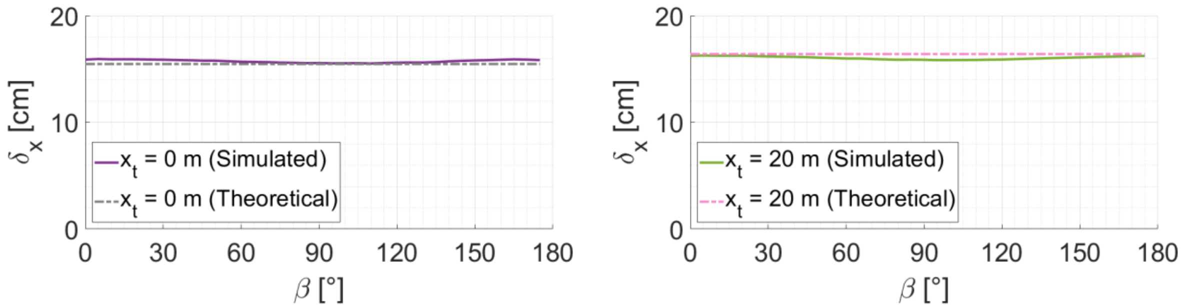
Figure 4.18 shows another flight path of interest from this simulation batch. It is the conical spiral flight path for which B is aligned with the LOS direction, i.e., $\beta = \psi_0 + 90°$. The blue flight path in Figure 4.17 is the opposite case: because $B = B_\perp$, it is perpendicular to the LOS.



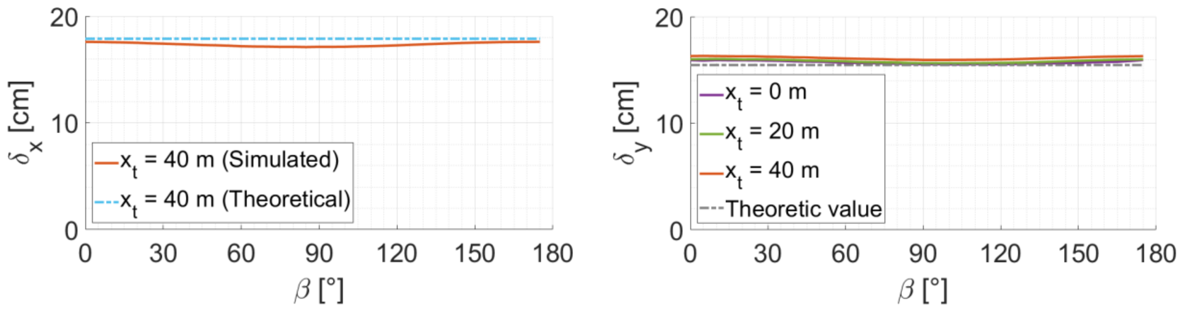
(a) (b)

Figure 4.18 – The conical spiral flight path for which B is aligned with the LOS direction ($\beta = 145°$).

4.3.3.2 Simulation results



(a) (c)



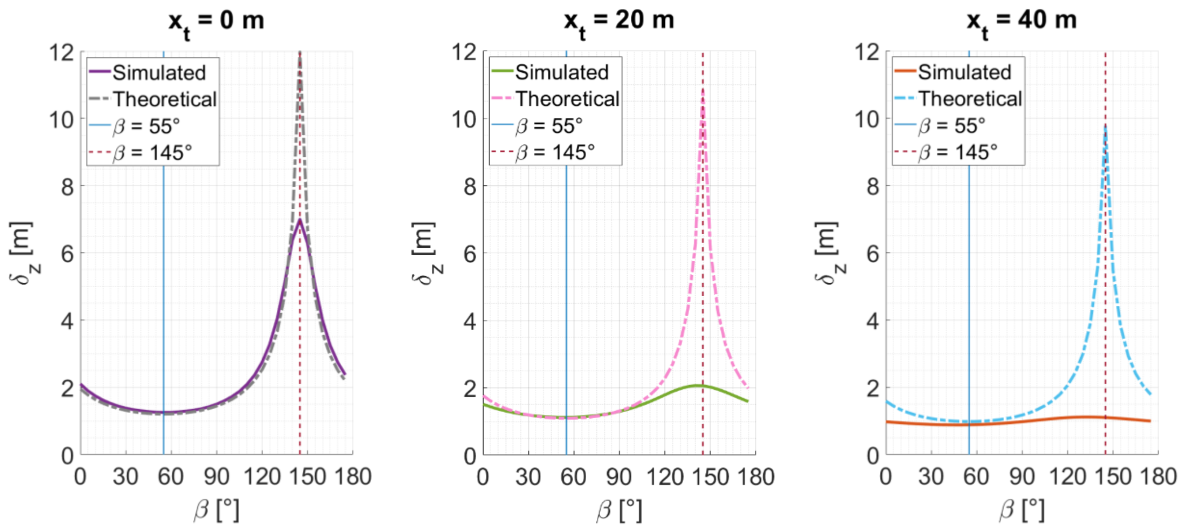
(b) (d)

Figure 4.19 – Simulated and theoretical ground resolution against β for the simulation batch with constant B: δ_x for the target at (a) $x_t = 0$ m, (b) $x_t = 20$ m, and (c) $x_t = 40$ m; (d) δ_y for all three targets.

Figure 4.19 shows the ground resolution (δ_x and δ_y) for the three isotropic targets against the tilt angle (β). Like in the last subsection, the curves are nearly constant because the parameters that impact δ_x and δ_y were kept constant. Once again, the radar positions at the near-range have more influence over the output resolution than those at the far-range. However, since the targets are located at the x-axis, the dynamic between near-range and far-range influence is not pronounced for δ_y. Therefore, only one theoretical value was calculated for δ_y (see Figure 4.19(d)).

On the other hand, three different theoretical values were calculated for δ_x, as shown in Figure 4.19(a-c). The theoretical values for the off-center targets were obtained using the equivalent look-angle (4.52). This approach provides a good δ_x estimate for all β.

Figure 4.20 shows the vertical resolution (δ_z) for the three isotropic targets as a function of β. Unlike before, δ_z varies significantly with β, even more so for the target at the center. For instance, in Figure 4.20(a), the simulated δ_z varies from 1.25 m to 7.00 m.



(a) (b) (c)

Figure 4.20 – Simulated and theoretical δ_z against β for the simulation batch with constant B. Each panel corresponds to the target at: (a) $x_t = 0$ m; (b) $x_t = 20$ m; (c) $x_t = 40$ m.

The best δ_z occurs when B is perpendicular to the LOS direction ($\beta = \psi_0$), represented by the blue flight path in Figure 4.17. Conversely, the worst vertical resolution occurs when B is aligned with the LOS direction ($\beta = \psi_0 + 90°$), such as the flight path depicted in Figure 4.18. In that case, the theoretical δ_z becomes equal to that of a single circle flight path because, in theory, the vertical spectral shift (4.26) is null. In reality, however, the wavenumber shift will still occur even if the tomographic aperture is aligned with the LOS direction.

The wavenumber shift will be more pronounced as the target moves away from the center. That is why the simulated curves in Figure 4.20(b,c) have much softer peaks than the theoretical curves. In Figure 4.20(b), the simulated δ_z varies from 1.12 m ($\beta = 55°$) to 2.06 m ($\beta = 140°$). In Figure 4.20(c), the simulated δ_z varies from 0.88 m ($\beta = 50°$) to 1.12 m ($\beta = 135°$). Note that the minimum and maximum values are slightly shifted to the left compared to Figure 4.20(a).

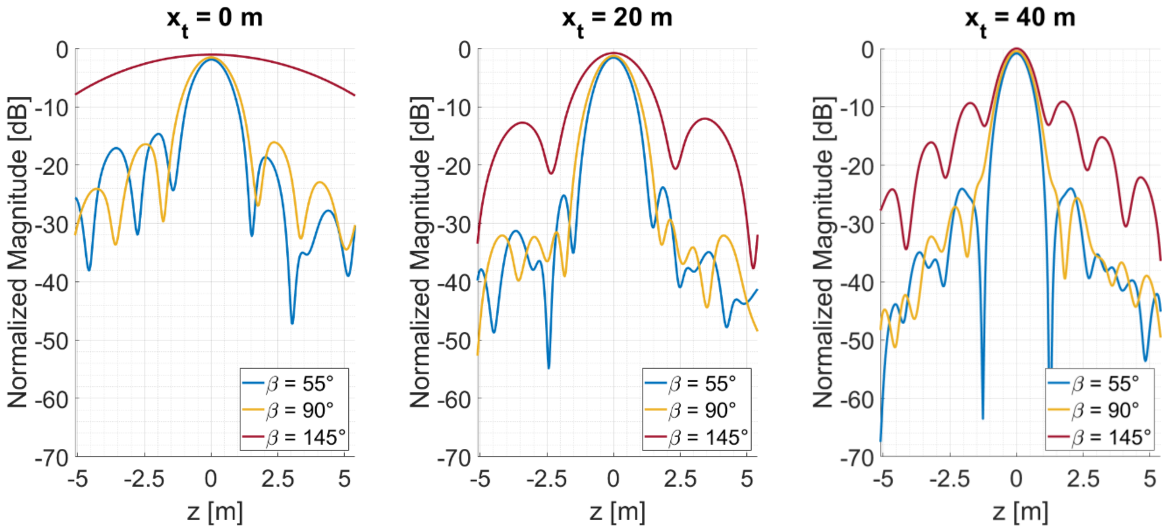
Once more, the radar positions at near-range exert more influence over the vertical resolution. That is why the theoretical curves of Figure 4.20(b,c) were determined using the equivalent bandwidth (4.53) and the equivalent look-angle (4.52). This approach provides a reasonable estimate when β is close to ψ_0. Meanwhile, in Figure 4.20(a), the theoretical curve matches the simulated curve well for all β, except at the peak. Table 4.4 presents two sets of RMSE values for δ_z: one taking all β into account; one for $30° \leq \beta \leq 90°$.

Table 4.4 – RMSE for the simulation batch with constant B, considering the near-range influence.

Target	**RMSE**			
	x [cm]	**y [cm]**	**z [m]**	
			$\forall\beta$	$30° \leq \beta \leq 90°$
$\boldsymbol{x_t} = 0$ m	0.25	0.27	0.04	0.05
$\boldsymbol{x_t} = 20$ m	0.36	0.38	0.80	0.01
$\boldsymbol{x_t} = 40$ m	0.54	0.69	1.07	0.13

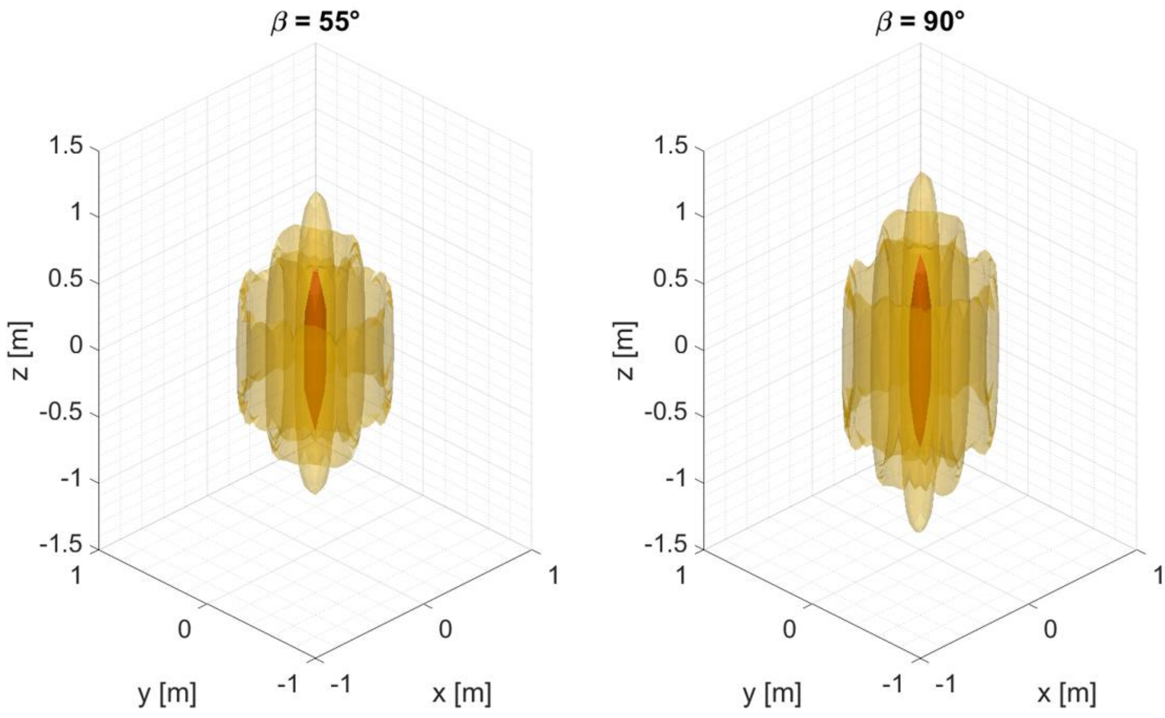
Figure 4.21 shows the normalized magnitude in dB at the z-axis for each target, comparing the responses for the three flight paths in Figure 4.17 and Figure 4.18. All curves in Figure 4.21 used the same reference value: the maximum amplitude obtained by the cylindrical flight path at the outermost target. Also, the curves were upsampled with a factor of 10.

Note that the resolution disparity decreases as the target moves away from the center, which is consistent with Figure 4.20. Nevertheless, the main lobe is always the narrowest for the conical flight path perpendicular to the LOS direction. On the other hand, the flight path aligned with the LOS direction has the worst performance in vertical resolution and PSLR. For the target at the center, this flight path has such a coarse vertical resolution that the main lobe is not fully contained within the imaged volume.



(a) (b) (c)

Figure 4.21 – Normalized magnitude in dB at the z-axis for three flight paths of the simulation batch with constant B: the conical spiral perpendicular to the LOS direction ($\beta = 55°$); the cylindrical spiral ($\beta = 90°$); the conical spiral aligned with the LOS direction ($\beta = 145°$). Each panel corresponds to the response of a different target: (a) $x_t = 0$ m; (b) $x_t = 20$ m; (c) $x_t = 40$ m.



(a) (b)

Figure 4.22 – Output images for the target at $x_t = 0$ m, obtained with two flight paths from the simulation batch with constant B: (a) the conical spiral perpendicular to the LOS direction ($\beta = 55°$) and (b) the cylindrical spiral ($\beta = 90°$). The -3 dB isosurfaces are opaque red, and the -13 dB isosurfaces are translucent yellow.

Figure 4.22 compares the responses of the conical flight path perpendicular to the LOS direction and the cylindrical flight path (see Figure 4.17), showing the -3 dB (red) and the -13 dB (yellow) isosurfaces for the target at the center. Again, the dB magnitudes were calculated using the peak amplitude of each volume as a reference. Comparing the volumes enclosed by the isosurfaces, the volume in Figure 4.22 (b) is the tallest, but their widths are about the same. This result is consistent with Figure 4.19 and Figure 4.20.

4.3.4 Spiral SAR vs. Multi-Circular SAR

This subsection shall investigate the hypothesis from Subsection 4.2.2.2 that Spiral SAR might not be affected by decorrelation issues when working with isotropic targets. We shall compare the performance of spiral flight paths with different numbers of turns with their equivalent multi-circular flight paths. Recall that a flight path with $N_t + 1$ circular tracks has the same sampling distance as a spiral flight path with N_t turns.

4.3.4.1 Simulation setup

This simulation batch took the spiral flight path with $B = B_\perp$ from the two previous batches, varying the number of turns (N_t). In addition, it only took the target at the center into account. Table 4.5 shows the parameters that changed compared to Table 4.1.

Table 4.5 – Parameters of the simulation batch investigating the number of turns/tracks.

		Parameter	Values	Units
Flight path	β	Tilt angle	55	degrees
	N_t	Number of turns	{2, 4, 6, 8, 10}	-
FFBP setup		Number of voxels at the output image	81 × 81 × 81	-
	$(x_t, 0, 0)$	Output image center	(0, 0, 0)	m
		First split into image blocks	1 × 1 × 3	-

4.3.4.2 Simulation results

The critical sampling distance for this simulation batch was already calculated in Subsection 4.2.2.2; it is $\Delta B_{\perp,c} = 5.4$ m. Therefore, since $B_\perp = 50$ m, the only case in this simulation batch for which the effective sampling distance ($\Delta B_\perp$) is below the critical value is $N_t = 10$. Figure 4.23 shows the wavenumber domain for the minimum and maximum N_t. Note that, because the gaps were practically closed, Figure 4.23(c) and Figure 4.23(d) are much more similar to each other than Figure 4.23(a) and Figure 4.23(b). Indeed, as N_t increases, the degree of coherence increases between a spiral flight path and its equivalent multi-circular flight path (see Figure 4.24).

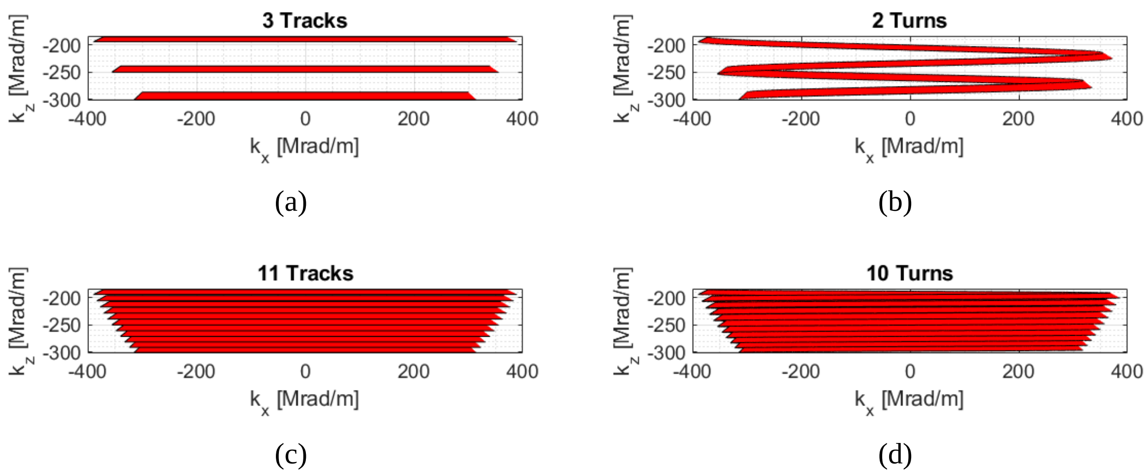


Figure 4.23 – Wavenumber domain for (a,c) Multi-Circular SAR acquisitions and (b,d) Spiral SAR acquisitions with (a,b) $N_t = 2$, and (c,d) $N_t = 10$.

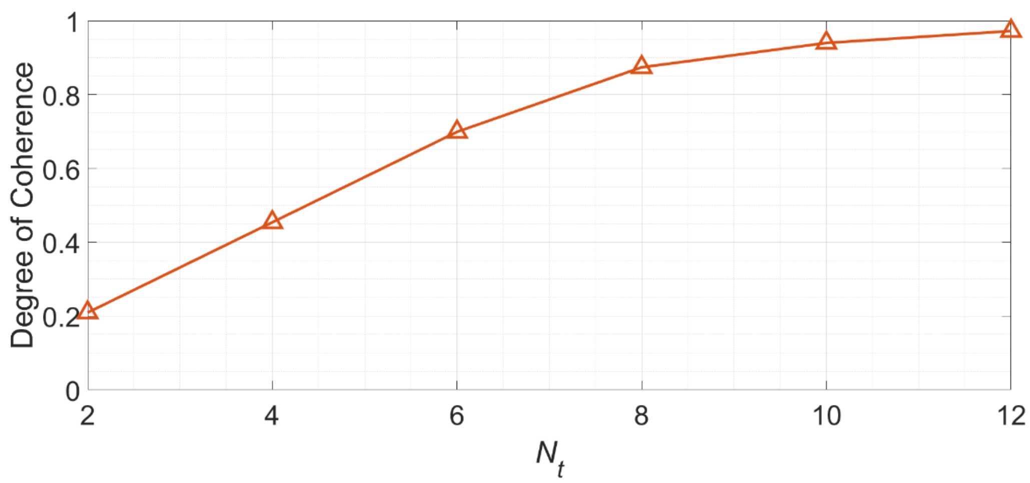


Figure 4.24 – Degree of coherence against N_t between a spiral flight path and its equivalent multi-circular flight path.

As N_t increases, more data are integrated onto each image voxel, so the amplitude of the main lobe increases. Therefore, to compare the performance for different values of N_t, the amplitude of each output image was divided by the number of turns or tracks, depending on the type of acquisition. Figure 4.25 shows the z-axis response for Multi-Circular SAR and Spiral SAR with different numbers of tracks and turns. Note that the Spiral SAR responses converge much faster than the Multi-Circular SAR responses.

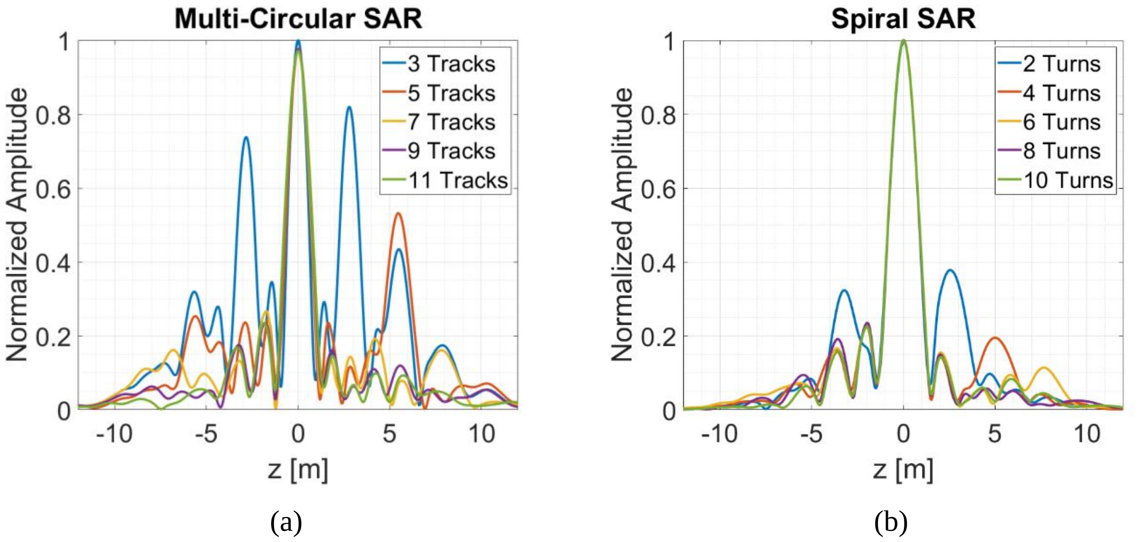


Figure 4.25 – Normalized amplitude at the z-axis for the simulation batch with variable N_t: (a) Multi-Circular SAR and (b) Spiral SAR. Each panel shows the entire image height.

Figure 4.26 highlights the main lobe of the z-axis response. Note that, for Spiral SAR, N_t does not affect the vertical resolution. On the other hand, for Multi-Circular SAR, it does. Although it may seem that the flight path with three circular tracks performs better because it has the narrowest main lobe, this same flight path has the strongest side lobes and the shortest height of ambiguity, $z_{2\pi} = 1.9$ m, calculated according to (4.32).

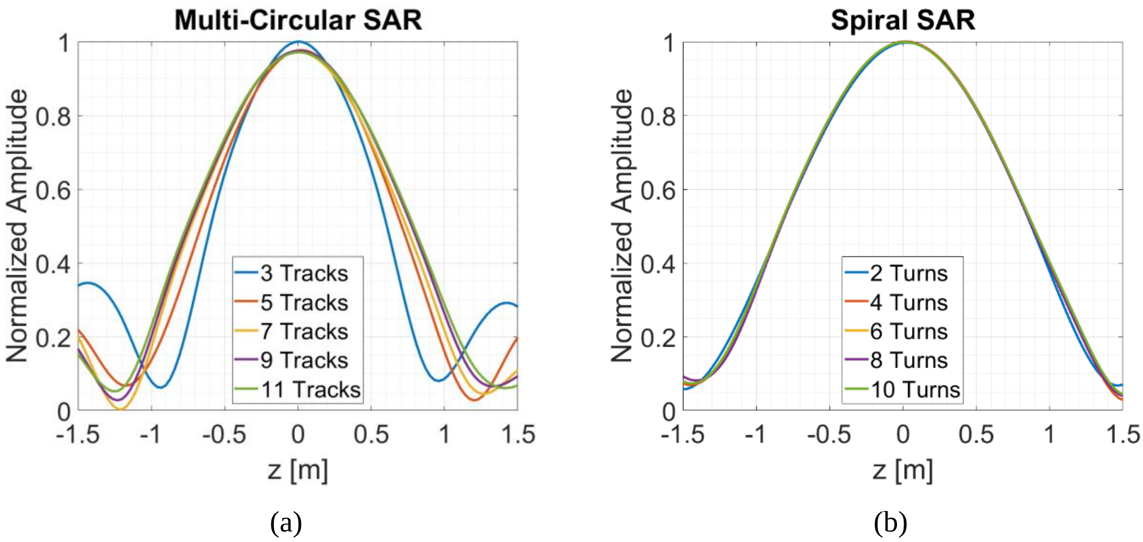
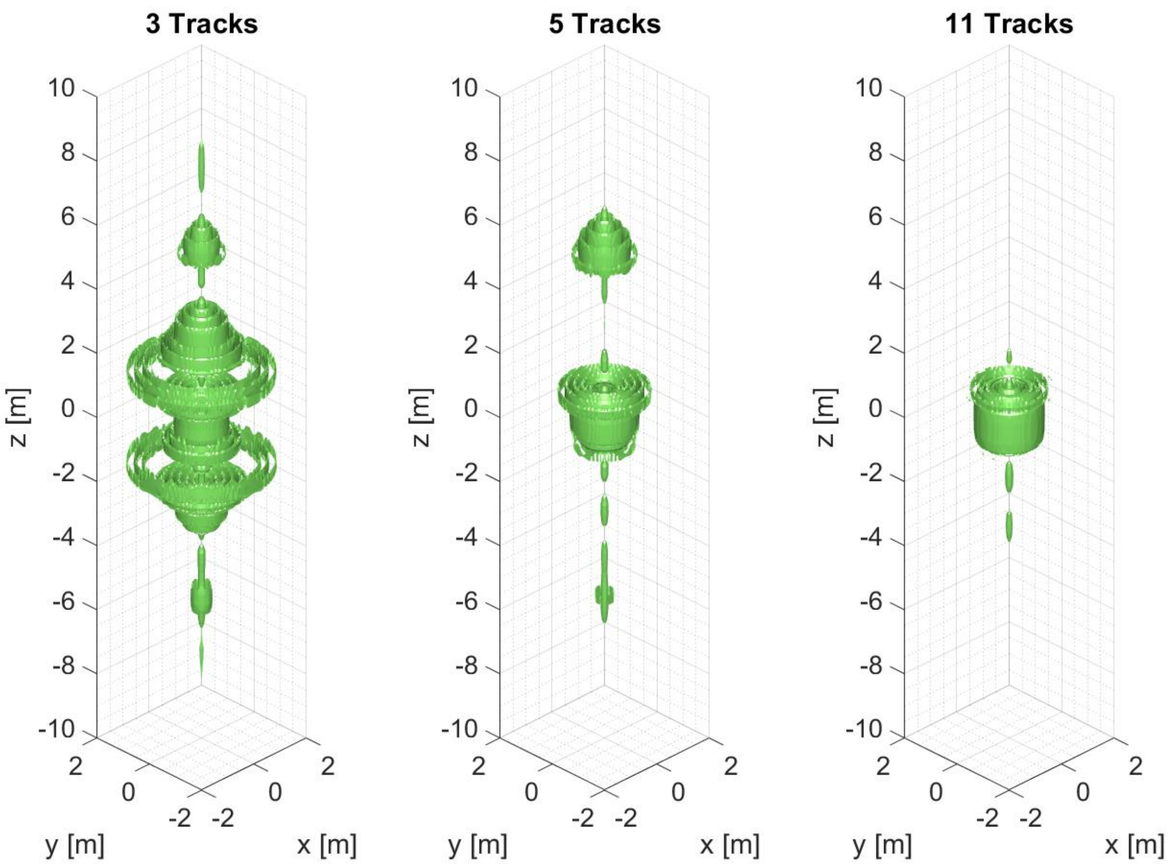


Figure 4.26 – Normalized amplitude at the z-axis for the simulation batch with variable N_t: (a) Multi-Circular SAR and (b) Spiral SAR. Each panel shows only the main lobe.

Figure 4.27 and Figure 4.28 show -20 dB isosurfaces of output images produced with different N_t by Multi-Circular SAR and Spiral SAR, respectively. Note that when $N_t = 4$, the main response resembles the one obtained with $N_t = 10$, but we can see a phantom at about

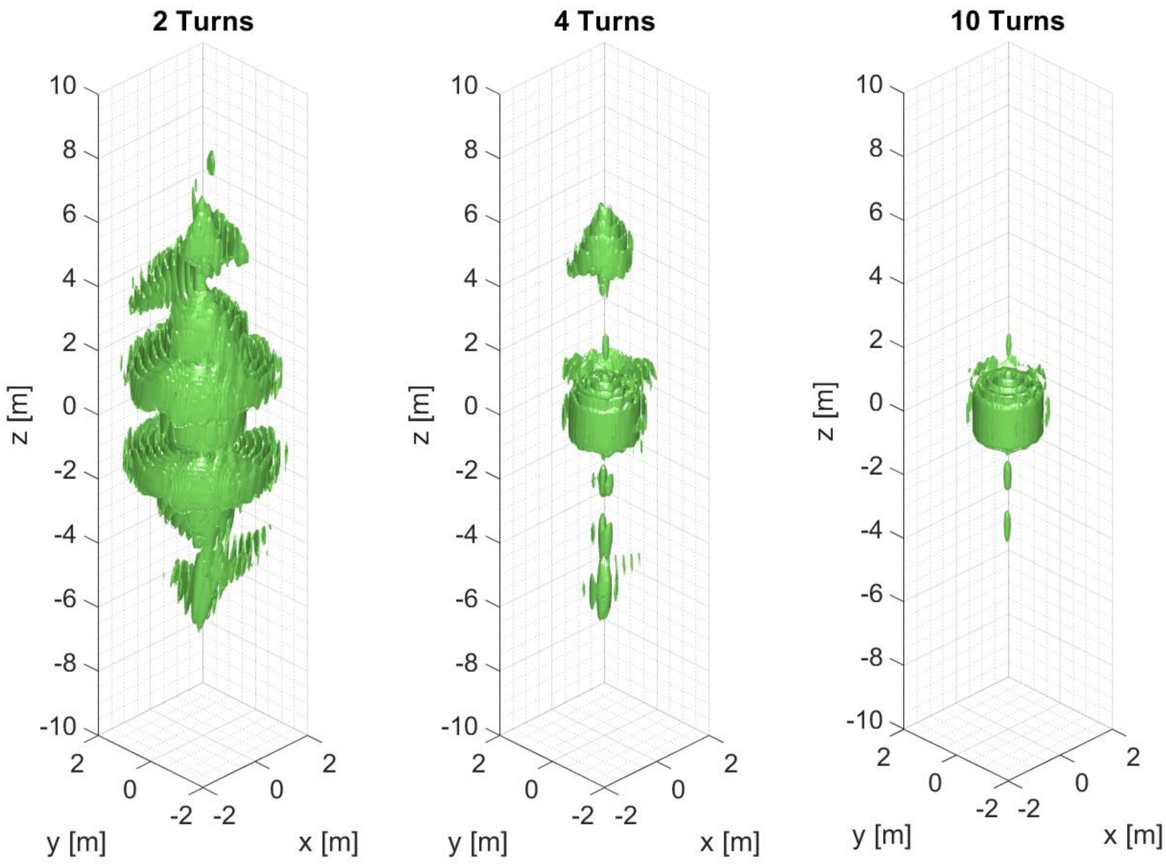
$z = 5$ m (see Figure 4.27(b,c) and Figure 4.28(b,c)). When $N_t = 2$, both output images are distorted by artifacts (see Figure 4.27(a) and Figure 4.28(a)). This result is probably due to the height of ambiguity being too short to the point of the phantoms interfering with the primary response. If the cause were decorrelation, the output would resemble the response of a singular circular flight path, as noted in [26]. However, that does not seem to be the case since the output of a singular circular flight path has an hourglass shape.



(a) (b) (c)

Figure 4.27 - Output images produced by multi-circular flight paths with (a) 3 tracks ($N_t = 2$), (b) 5 tracks ($N_t = 4$), and (c) 11 tracks ($N_t = 10$). The -20 dB isosurfaces are opaque green.

Finally, Table 4.6 shows the simulated flight times for the different configurations in this simulation batch. On average, Spiral SAR was 1 min 56 s faster than Multi-Circular SAR. That difference is significant because the drone's autonomy is quite limited (less than 20 min depending on the payload [61]). The advantage of a lower flight time is clearer for the scenario with $N_t = 10$. In that case, the flight time surpasses 20 min for Multi-Circular SAR but is kept below that threshold for Spiral SAR.



(a) (b) (c)

Figure 4.28 – Output images produced by spiral flight paths with (a) 2 turns ($N_t = 2$), (b) 4 turns ($N_t = 4$), and (c) 10 turns ($N_t = 10$). The -20 dB isosurfaces are opaque green.

Table 4.6 – Simulated flight time for Multi-Circular SAR and Spiral SAR with different N_t.

SAR Type	Simulated Flight Time				
	$\boldsymbol{N_t = 2}$	$\boldsymbol{N_t = 4}$	$\boldsymbol{N_t = 6}$	$\boldsymbol{N_t = 8}$	$\boldsymbol{N_t = 10}$
Multi-Circular	5 min 39 s	9 min 25 s	13 min 12 s	16 min 58	20 min 44 s
Spiral	3 min 45 s	7 min 31 s	11 min 16 s	15 min 1 s	18 min 47 s

4.3.5 Discussion

This section demonstrated that conical flight paths could perform better than cylindrical flight paths, depending on the tilt angle. The best-case scenario occurs when the tomographic aperture is perpendicular to the LOS direction. Conversely, the worst-case scenario occurs when the tomographic aperture is aligned with the LOS direction. This section also confirmed that the vertical resolution is a function of the effective tomographic aperture.

Therefore, a cylindrical flight path could match the performance of the best-case scenario if we increased the height difference between the top and base of the spiral. However, that could become an issue when working with drones due to the low flight ceiling required for most operations.

Furthermore, this section validated the vertical resolution equation (4.25-4.27) proposed in Section 4.2. It worked well for a target at the center, producing low RMSE values. In addition, this section offered a method for estimating the ground and vertical resolutions for off-center targets. This method came about after observing that near-range radar positions have more influence over the output result. It provided good estimations, with low RMSE, for tilt angles close to the midpoint look-angle.

Moreover, this section showed that the advantage of Spiral SAR over Multi-Circular SAR is twofold: Spiral SAR converges faster to a satisfactory response as the number of turns increases; Spiral SAR requires lower flight times. When combining those two advantages, Spiral SAR becomes a powerful tool for working with limited autonomy drones.

On the other hand, the hypothesis that decorrelation does not affect Spiral SAR performance could not be confirmed. Because the simulations mimicked a drone operating on the P-band, the dimensions of the problem were too small to reproduce a scenario where decorrelation might occur. Indeed, Ponce *et al.* [26] observed decorrelation for an aircraft operating on the L-band and a spacing of over 100 m between tracks. With that in mind, the hypothesis needs further investigation.

The results in this section comprised 63 different flight paths and 172 3D SAR images in total. If not for the FFBP algorithm, these analyses would have been impractical. Moreover, to the best of the author’s knowledge, this is the first study that investigated the effect of geometric parameters on the output image. Sego *et al.* [17] had studied different kinds of spiral flight paths, but with only one configuration for each kind.

4.4 FLIGHT PATH DESIGN

Section 4.2 presented expression (4.27) for calculating the vertical resolution of spiral flight paths. Section 4.3 validated that expression and further investigated the performance of spiral flight paths using simulation results. This section provides a methodology for designing a spiral flight path that uses expression (4.27) and has two main components: design constraints and a trade-off. The methodology is an original contribution from the author.

In addition, this section presents processing results from actual SAR data to verify the proposed trade-off. The SAR data were processed with the FFBP algorithm (see Section 3.2) on an Intel(R) Core(TM) i7-7700 CPU (3.60 GHz) with 64 GB RAM.

4.4.1 Design Constraints

When designing a spiral flight path for any drone-borne SAR system, the primary constraints are the maximum and minimum height, the drone's speed and autonomy, and the area constantly illuminated by the radar. The maximum height, or flight ceiling, is stipulated by aviation regulation agencies and is 120 m (400 ft) for most operations [58]–[60], as mentioned previously in Section 4.3. The minimum height might be adjusted according to a tall construction in the scene, or it can simply be arbitrary. For the drone-borne SAR system of Section 1.1, the maximum speed is 18 m/s with no wind, and the autonomy is less than 20 min with a full payload [61]. However, the flight path design should consider a margin for these constraints. The speed margin is to obtain high-quality images – the slower the drone flies, the better the output image quality. The autonomy margin is for performing ground operations before and after the flight.

The autonomy provides an upper bound for the number of turns in the spiral flight path. In addition, the critical sampling distance could determine a lower bound (see Subsection 4.2.2.2). The purpose of the lower bound would be to avoid decorrelation, but this effect needs further investigation (see Subsection 4.3.4).

The last constraint is the area constantly illuminated by the radar, which is determined by the flight path. Let the radius of constant illumination (ρ_{ci}) be the radius of the area always seen by the radar throughout the flight path. Then, ρ_{ci} is defined by two limiting conditions: the maximum radius (ρ_{max}) with the minimum height (z_{min}) at the base of the flight path; the minimum radius (ρ_{min}) with the maximum height (z_{max}) at the top of the flight path.

Figure 4.29 illustrates the first condition, which results in the following expressions:

$$\rho_{ci}^{base} \leq \frac{z_{min}}{\tan(\theta_{far})} - \rho_{max} \tag{4.54}$$

$$\theta_{far} = \theta_{axis} - \frac{\theta_{3dB}}{2} \tag{4.55}$$

where θ_{far} is the far-range depression angle, θ_{axis} is the depression angle of the antenna axis, and θ_{3dB} is the antenna elevation beamwidth.

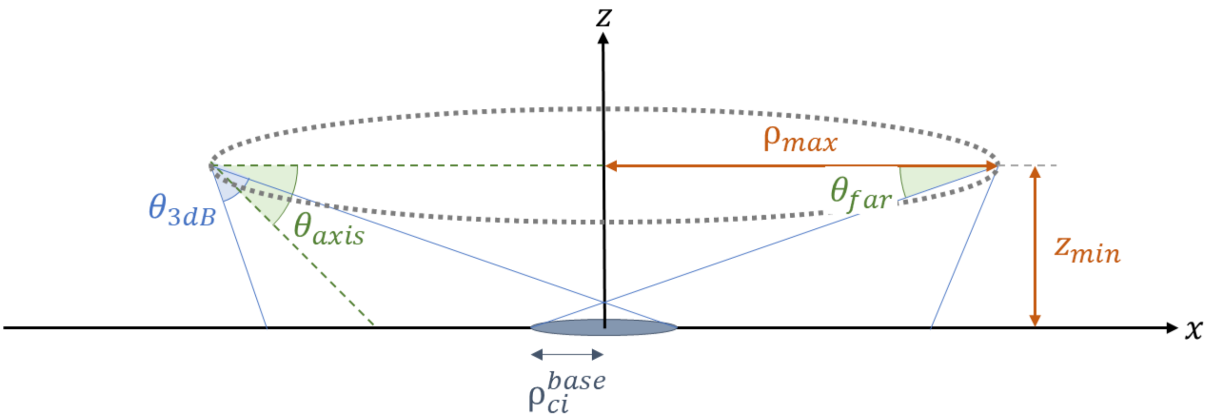


Figure 4.29 – The first limiting condition for calculating the radius of constant illumination: the maximum radius (ρ_{max}) and the minimum height (z_{min}) at the base of the flight path.

Figure 4.30 shows the second condition, which produces the expressions:

$$\rho_{ci}^{top} \leq \rho_{min} - z_{max}\tan(\theta_{near}) \tag{4.56}$$

$$\theta_{near} = 90° - \theta_{axis} - \frac{\theta_{3dB}}{2} \tag{4.57}$$

where θ_{near} is the near-range look-angle.

Finally, the radius of constant illumination is given by:

$$\rho_{ci} = \min(\rho_{ci}^{base}, \quad \rho_{ci}^{top}) \tag{4.58}$$

4.4.2 The Trade-Off

In Subsection 4.3.2.2, it was implied that there is a trade-off between vertical and ground resolutions. This trade-off is made clear by the dependence of sine and cosine in equations (4.4) and (4.5). Likewise, this subsection shall present another trade-off between the vertical resolution (δ_z) and the radius of constant illumination (ρ_{ci}). This trade-off must be taken into account when designing spiral flight paths for drones.

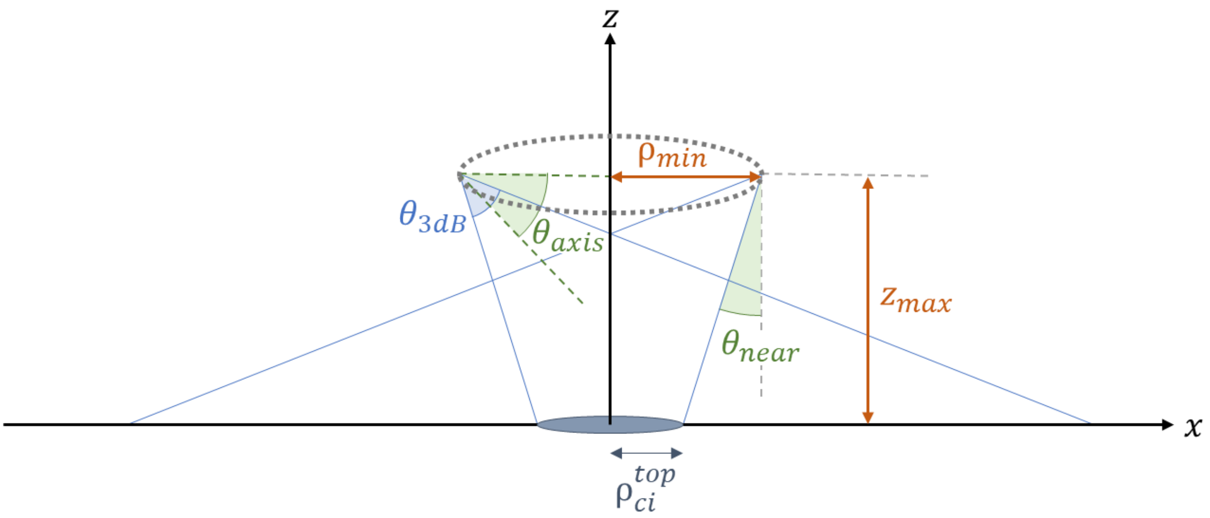


Figure 4.30 – The second limiting condition for calculating the radius of constant illumination: the minimum radius (ρ_{min}) and the maximum height (z_{max}) at the top of the flight path.

From the last subsection, we know that the minimum and maximum heights (z_{min} and z_{max}) are two of the main design constraints. Suppose we fix these two quantities and choose an arbitrary value for the maximum radius (ρ_{max}). Also, suppose we vary the minimum radius (ρ_{min}) by making:

$$\rho_{min} = \rho_{max} - \Delta\rho \tag{4.59}$$

where $\Delta\rho$ is the radius variation. Then, the trade-off will be made clear when we plot δ_z against ρ_{ci}, as shown in Figure 4.31: δ_z improves (decreases) as ρ_{ci} declines.

Figure 4.31 was drawn using $z_{max} = 114$ m, $z_{min} = 84$ m and $\rho_{max} = 118.5$ m. Also, $\Delta\rho$ varied from 0 to 60 m in increments of 1 m, with $\Delta\rho = 0$ m corresponding to the cylindrical flight path (green square) at the right end of the curve. As $\Delta\rho$ increases, the tomographic aperture (B) increases, making both δ_z and ρ_{ci} decrease. What limited ρ_{ci} was the near-range look-angle, equal to $\theta_{near} = 22.2°$ for the P-band antenna. The tomographic aperture was $B = 30$ m for the cylindrical flight path, $B = 36.6$ m for the conical flight path with $\beta = 55°$ (yellow circle) and $B = 60$ m for the conical flight path with $\beta = 30°$ (blue triangle).

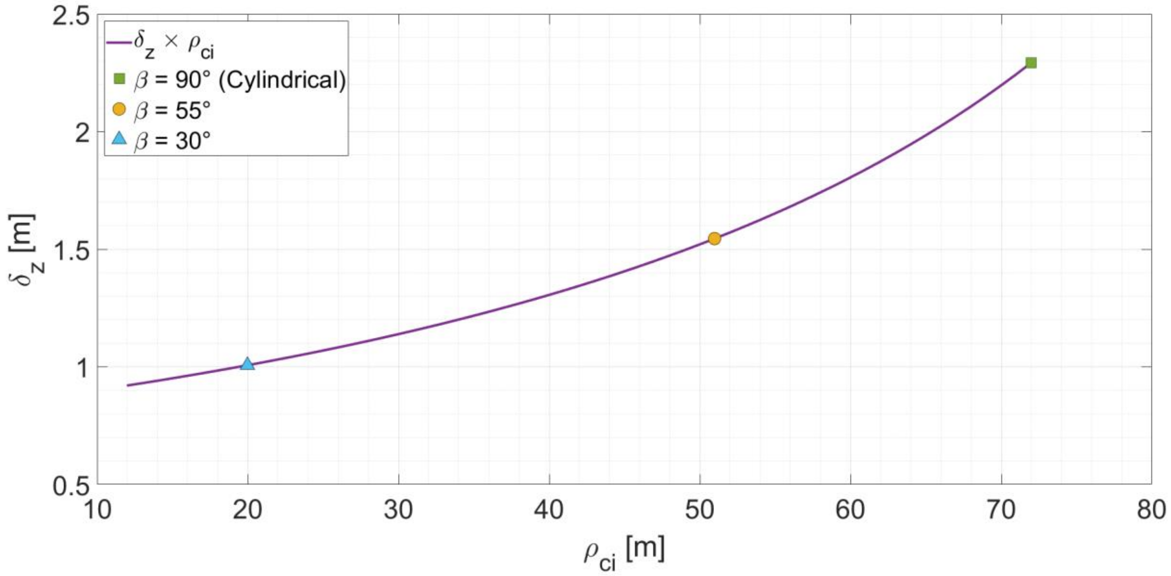


Figure 4.31 – Trade-off curve: $\delta_z \times \rho_{ci}$. Flight paths of interest: cylindrical spiral, $\beta = 90°$ (green square); conical spiral with $\beta = 55°$ (yellow circle); conical spiral with $\beta = 30°$ (blue triangle).

In Figure 4.31, the conical flight path with $\beta = 55°$ provides a good compromise solution. From the cylindrical flight path to that point, δ_z improved 1.5 times as ρ_{ci} decreased 1.4 times. On the other hand, when we compare the cylindrical flight path with the conical flight path with $\beta = 30°$, δ_z improved 2.3 times as ρ_{ci} decreased 3.4 times.

To generate different design alternatives, we could repeat this procedure for multiple values of ρ_{max} or vary z_{min} and z_{max}. Then, we would choose the best option for each application, taking the trade-off between δ_z and ρ_{ci} into account.

4.4.3 The Flight Campaign

A flight campaign to compare different flight paths from Figure 4.31 took place in Paulínia, São Paulo, Brazil, on July 30, 2021. The drone-borne SAR system (see Section 1.1) flew over an experimental farm with different crops and performed the cylindrical flight path (see Figure 4.32) and the conical flight path with a tilt angle $\beta \approx 55°$ (see Figure 4.33).

In Figure 4.32 and Figure 4.33, note that the flight paths are not smooth spirals but sequences of line segments. The reason is the limited number of waypoints for programming the drone's flight path. Consequently, the radius variation is not null for the cylindrical flight path but 5.6 m. Nevertheless, this radius variation is still quite different from the conical flight path, which equals 20.7 m. Note that this difference is quite evident in Figure 4.32 and Figure 4.33.

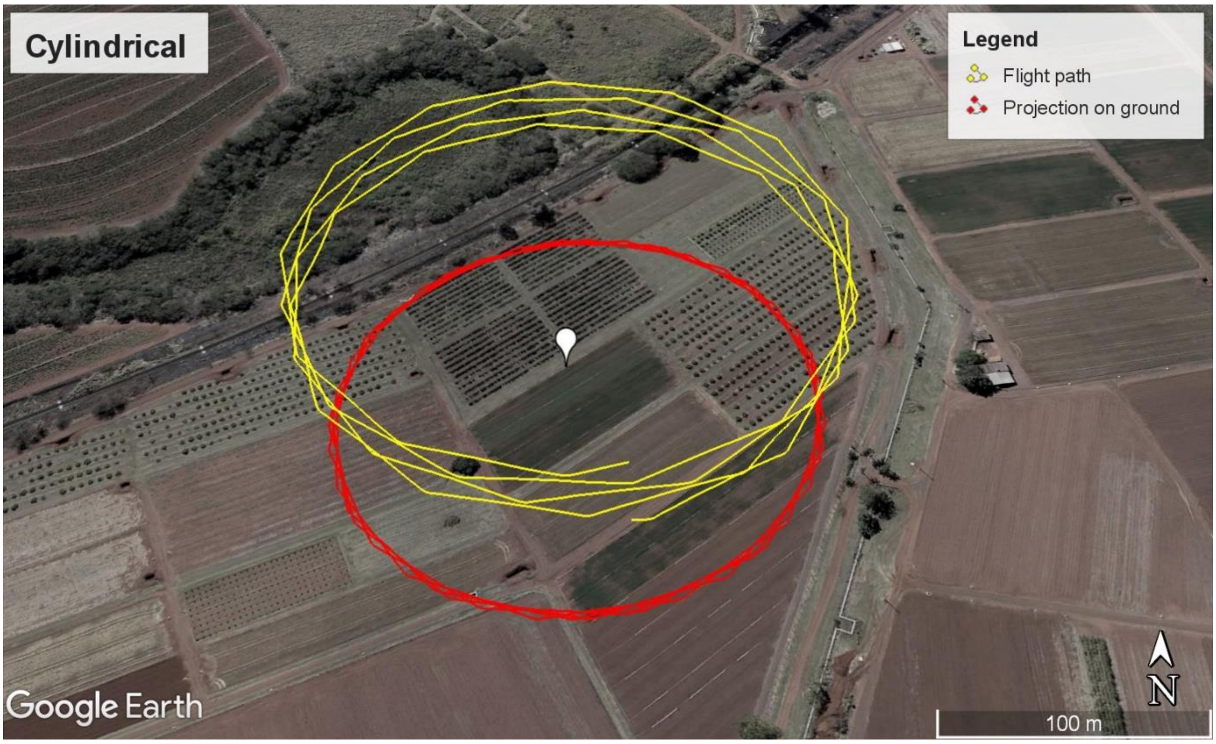


Figure 4.32 – Google Earth picture of the cylindrical flight path over the imaged area. The white marker, the yellow line, and the red line respectively indicate the position of the quad-corner reflector, the flight path, and its projection on the ground.

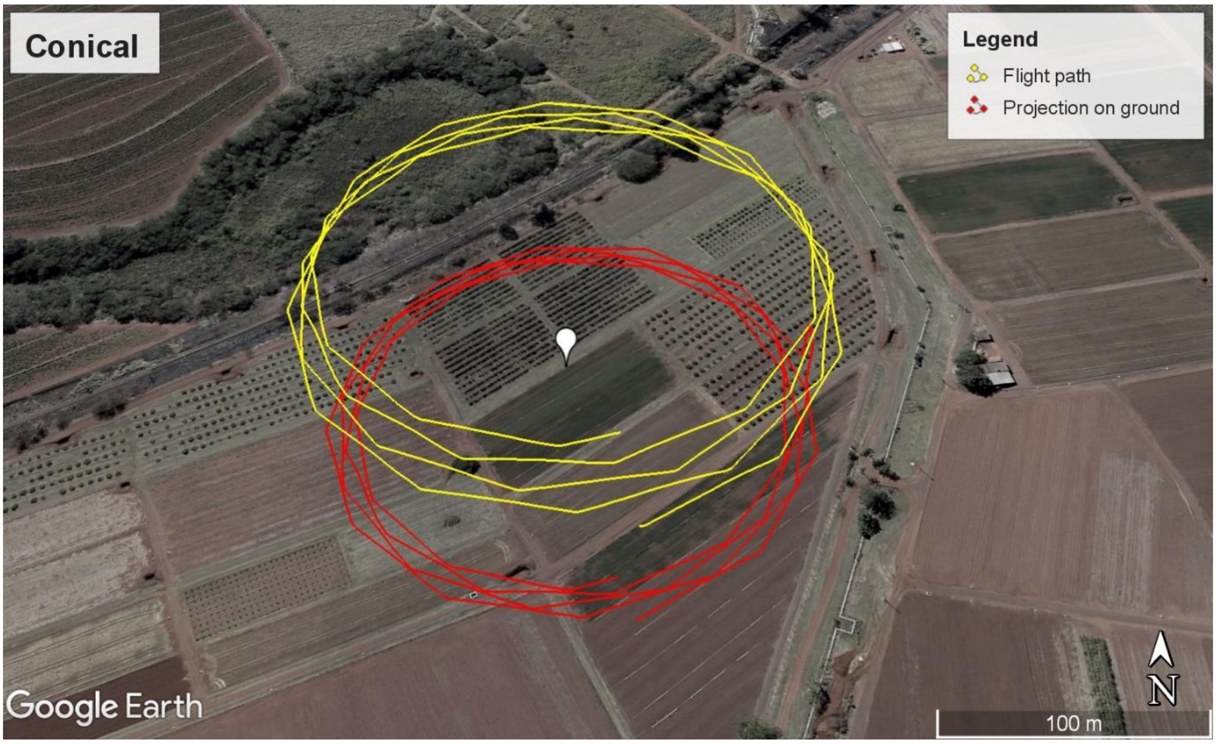


Figure 4.33 – Google Earth picture of the conical flight path over the imaged area. The white marker, the yellow line, and the red line respectively indicate the position of the quad-corner reflector, the flight path, and its projection on the ground.

Moreover, a quad-corner reflector was used to evaluate the performance of each flight path in terms of vertical resolution. A white marker indicates the reflector position in Figure 4.32 and Figure 4.33. The quad-corner reflector is made of four trihedral corner reflectors with sides of 50 cm, as depicted in Figure 4.34.

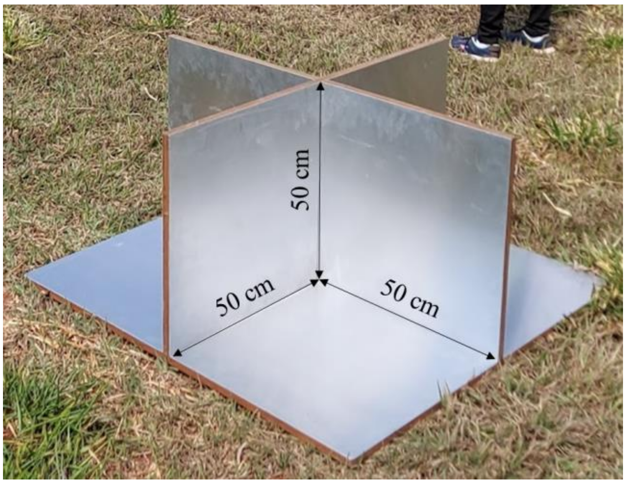


Figure 4.34 – The quad-corner reflector and its dimensions

Table 4.7 shows the parameters for both flight paths. The mean drone speed was determined after removing spurious values.

Table 4.7 – Radar acquisition parameters and parameters of the spiral flight paths

	Parameter	Spiral		Units
		Cylindrical	**Conical**	
Flight paths	Minimum radius	112.6	98.1	m
	Maximum radius	118.2	118.8	m
	Minimum height	83.2	84.4	m
	Maximum height	114.3	113.0	m
	Number of turns	4	4	-
	Mean drone speed	6.94	6.85	m/s
	Number of radar root positions	67,355	63,457	-
Radar	Wavelength	70.54		cm
	Bandwidth before range compression	50		MHz
	Pulse repetition interval	6.642		ms
	Range sampling interval	2.456		m

4.4.4 The Digital Terrain Model (DTM)

From Sections 2.4 and 3.2, the BP and FFBP algorithms require height information for each pixel/voxel of the imaged area/volume. Although SRTM data is freely available [28], it might not be accurate enough to obtain a well-focused image with BP and FFBP algorithms. Therefore, we can first process the SAR data using SRTM data and then apply one of the techniques in Section 4.1 to find a more refined digital terrain model (DTM).

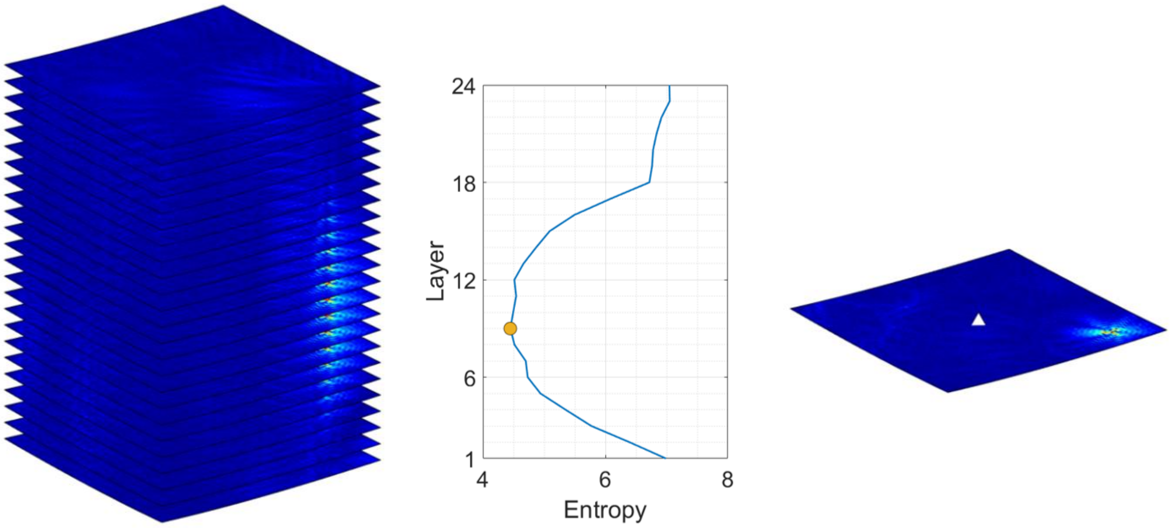
This section is divided into two steps: creating and validating the DTM. The first step requires a 3D image encompassing the whole scene. The second step needs 2D images to compare the performances of the SRTM data and the refined DTM. Table 4.8 provides the FFBP setup parameters for both steps.

Table 4.8 – FFBP setup parameters for creating and validating the Digital Terrain Model (DTM).

Parameter		Values	Units
3D Image	Output image dimensions	120 × 120 × 7.2	m^3
	Output image resolution	10 × 10 × 30	cm^3
	Output image center	(0,0,0)	m
	Subapertures combined at each recursion (L)	4	-
	First split into image blocks	4 × 4 × 1	-
2D Images	Output image dimensions	100 × 100	m^2
	Output image resolution	10 × 10	cm^2
	Output image center	(0,0,0)	m
	Subapertures combined at each recursion (L)	4	-
	First split into image blocks	4 × 4 × 1	-

4.4.4.1 Creating the Digital Terrain Model

The cylindrical flight path was chosen for building the DTM since it provides a greater coverage area than the conical flight path. First, a 3D image was processed with the FFBP algorithm using SRTM data. Next, the output 3D image was divided into 12 × 12 m^2 blocks. Then, each image block was separated into 2D layers, and the entropy was calculated layer by layer. Entropy is a measure of focus: the lower the entropy, the more focused the image is [23]. Therefore, the layer with the minimum entropy was chosen for each image block. Finally, the mean position of each chosen layer was calculated, and its coordinates were incorporated into the DTM. This process is illustrated in Figure 4.35.



(a) (b) (c)

Figure 4.35 – Building the DTM: (a) 2D layers of a 12 × 12 m² image block; (b) the entropy of each layer, with the minimum value indicated by a yellow circle; (c) the layer with the minimum entropy, with its mean position indicated by a white triangle.

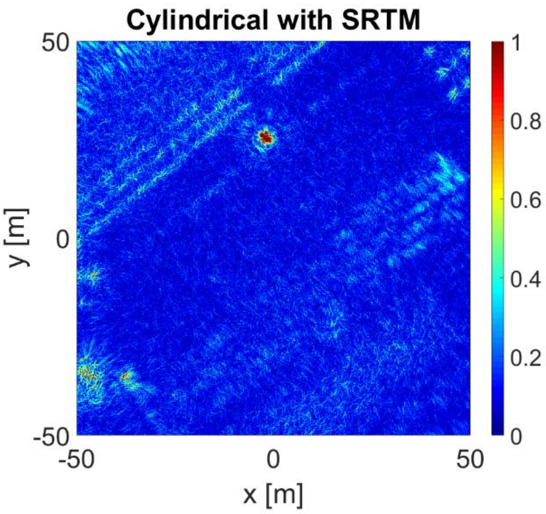
4.4.4.2 Validating the Digital Terrain Model

Figure 4.36 compares 2D images processed using: (a,c) SRTM data and (b,d) the refined DTM. Overall, the images produced with the DTM are sharper, especially where artificial objects are located, such as the quad-corner reflector at (-2, 25.4) m, a standard corner reflector at (15, -22) m, and a concrete structure at (-37.5, -34.5) m. In addition, the northwest crop is more focused, and the southeast crop has stronger contrast for the images produced with the DTM. The only exception is the crop in the center-east, which appears to lose contrast.

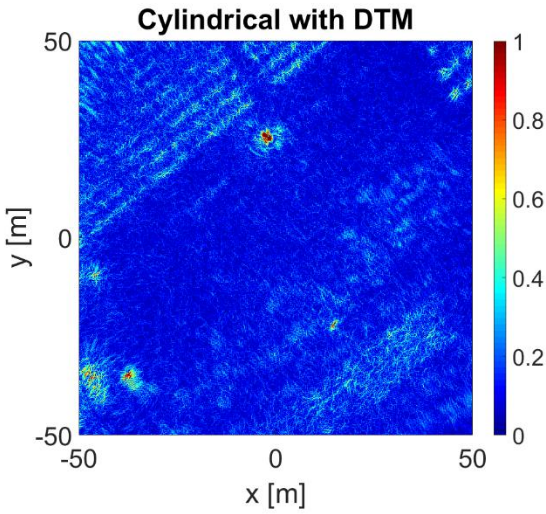
Note how the quad-corner reflector is more focused on the images produced with the conical flight path (Figure 4.36(c,d)) than on the images produced with the cylindrical flight path (Figure 4.36(a,b)). In addition, two vehicles were on the scene during the conical flight path but not during the cylindrical flight path. They appear in Figure 4.36(c,d) at around (-10, -45) and (-4, -40) m.

Note how the conical flight path loses focus near the image corners, i.e., when the coordinates approach $(x, y) = (\pm 50, \pm 50)$ m. This response confirms the expected behavior regarding the radius of constant illumination (ρ_{ci}). From Figure 4.31, ρ_{ci} should be 51 m for the conical flight path and 72 m for the cylindrical flight path. The results presented in Figure 4.36 are consistent with these values since the distance from the image center to the image corners is $50\sqrt{2}$ m (70.7 m). In addition, the limited number of waypoints makes the flight path radius vary for the cylindrical flight path (see Figure 4.32), which could slightly degrade ρ_{ci}.

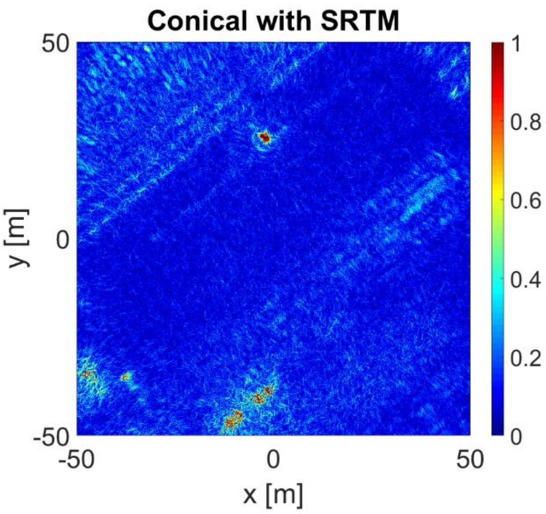
Because the radar cross-section of the quad-corner reflector is far superior to any other target, the amplitude of each image was saturated at a specific value before normalizing. The saturation value was equal to the mean amplitude plus ten times the standard deviation of each image.



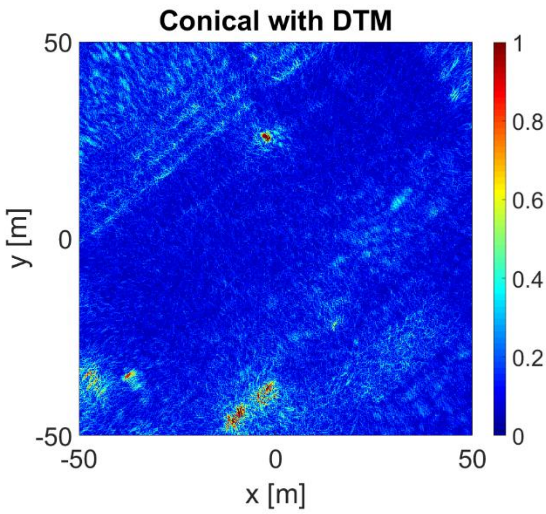
(a)



(b)



(c)



(d)

Figure 4.36 – Output 2D images for the SAR data acquired with (a,b) a cylindrical flight path; (c,d) a conical flight path. The SAR data were processed using: (a,c) SRTM data; (b,d) the refined DTM.

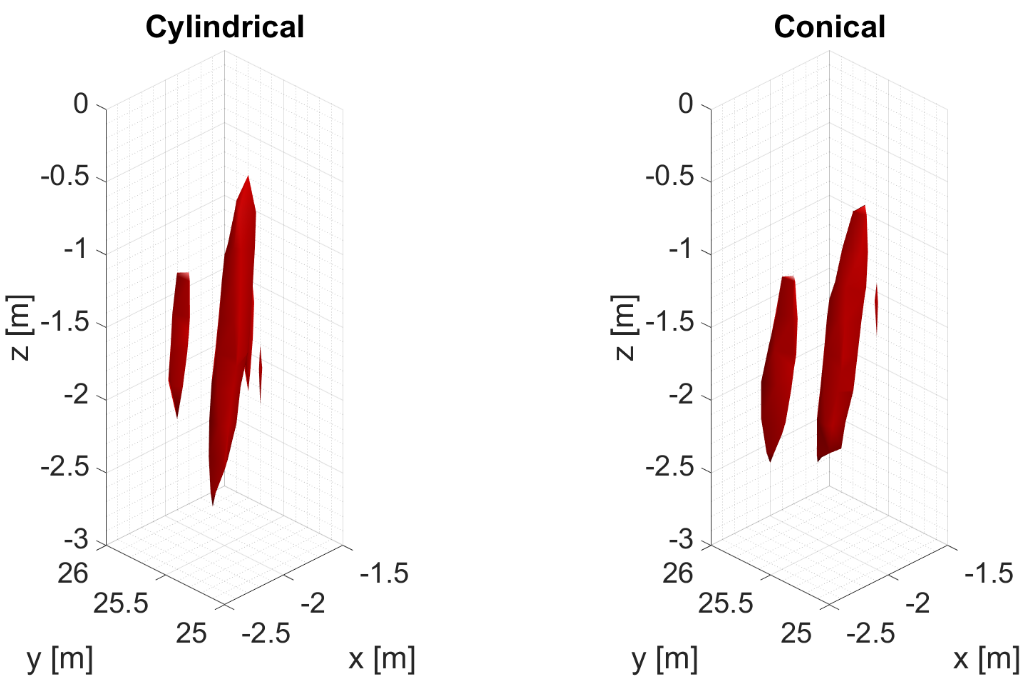
4.4.5 Output Resolution

The last subsection confirmed that the conical flight path has a smaller radius of constant illumination than the cylindrical flight path. Nevertheless, we still need to verify the vertical resolution to validate the trade-off presented in Subsection 4.4.2. For that purpose, 3D images enclosing the quad-corner reflector were processed with the FFBP algorithm. Table 4.9 shows the FFBP setup parameters.

Table 4.9 – FFBP setup parameters for the images encompassing the quad-corner reflector.

Parameter	Values	Units
Output image dimensions	6.4 × 6.4 × 6.4	m³
Output image resolution	10 × 10 × 10	cm³
Output image center	(-2, 25.5, 0)	m
Subapertures combined at each recursion (L)	4	-
First split into image blocks	1 × 1 × 1	-

The quad-corner reflector is neither an isotropic nor an omnidirectional target. Although its coverage is nearly 360°, its echo pattern has angles of maximum and minimum response [62], [63]. As a consequence, this amplitude modulation produces unwanted artifacts. Figure 4.37 shows the -3dB isosurfaces for the cylindrical and the conical flight paths. Since the quad-corner reflector is located at (-2, 25.4) m, an ideal response would contain only the main lobe found at that position.



(a) (b)

Figure 4.37 – Output 3D images for the quad-corner reflector with unwanted artifacts. The SAR data were acquired with: (a) the cylindrical flight path; (b) the conical flight path. The red curves are -3dB isosurfaces.

Thankfully this effect can be compensated by equalizing the input SAR data. First, data corresponding to the quad-corner reflector was determined for each radar position, using a nearest-neighbor interpolation to find the respective range bins. Then, the resulting signal was filtered using two moving averages with sliding windows of lengths 31 and 551 pixels. Next, each column of the SAR data matrix was divided by the filtered signal through a point-by-point division. Finally, the equalized SAR data were processed with the FFBP algorithm using the same setup parameters (see Table 4.9). Figure 4.38 shows the output 3D images after equalization. Admittedly, the equalizing function could be polished further, but the improvement was already significant.

Note how the responses of both Figure 4.37 and Figure 4.38 are slightly tilted. This effect is consistent with the simulation results presented in Subsection 4.3.2 (see Figure 4.16).

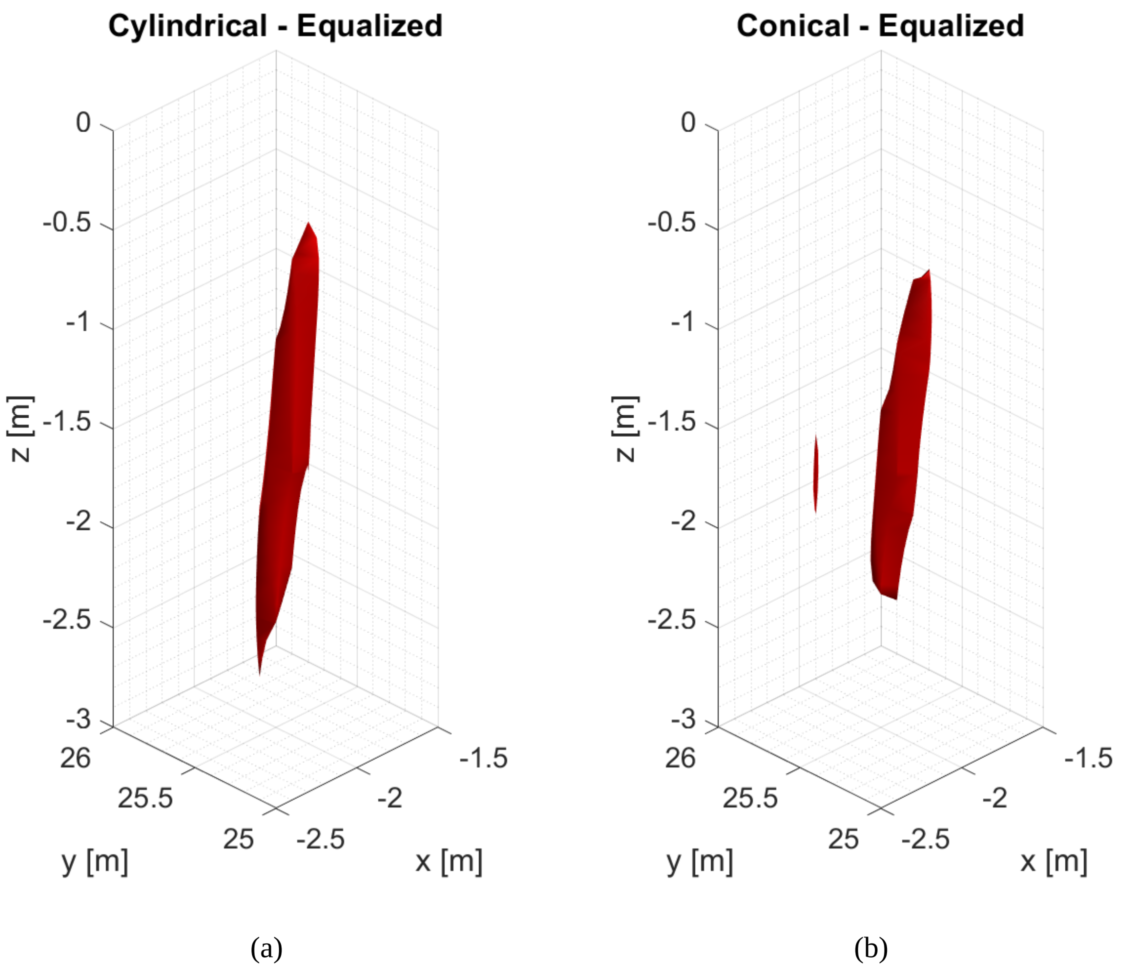


Figure 4.38 – Output 3D images for the quad-corner reflector after equalizing the SAR data for removing unwanted artifacts. The SAR data were acquired with: (a) the cylindrical flight path; (b) the conical flight path. The red curves represent -3dB isosurfaces.

Figure 4.38 exposes the difference in vertical resolution between the two flight paths, making it clear that the conical flight path performed better than the cylindrical in this respect. The vertical resolutions were 1.76 m for the conical flight path and 2.33 m for the cylindrical. Because the quad-corner is not an isotropic target, these values differ from the theoretical ones (see Table 4.10). However, the ratio between the cylindrical and conical resolutions is almost the same in each case: 1.32 for the actual values; 1.30 for the theoretical values considering a target at the center; 1.25 for the theoretical values considering a target at (-2, 25.4) m. Therefore, the results confirm the expected relative performance of the flight paths regarding vertical resolution. Thus, we can validate the trade-off proposed in Subsection 4.4.2.

Furthermore, the results confirm the expected ground resolutions, as shown in Table 4.10. Note that the actual values of δ_y are closer to the theoretical values calculated for a target at the center than for a target at (-2, 25.4) m. Either this result is a side effect of equalizing the SAR data, or the near-range effect is not as expressive in actual data. A more conclusive answer would require further investigation.

Table 4.10 – Resolutions in space for each flight path, comparing the actual and theoretical values calculated for a target at the center (0, 0) m and a target at (-2, 25.4) m.

Resolution	**Cylindrical Flight Path**			**Conical Flight Path**		
	Theoretical Value		**Actual Value**	**Theoretical value**		**Actual Value**
	(0, 0)	**(-2, 25.4)**		**(0, 0)**	**(-2, 25.4)**	
$\boldsymbol{\delta_x}$ **[cm]**	16.6	16.8	16.6	17.1	17.3	17.3
$\boldsymbol{\delta_y}$ **[cm]**		18.8	17.4		19.6	17.1
$\boldsymbol{\delta_z}$ **[m]**	1.98	1.68	2.33	1.52	1.34	1.76

4.4.6 Discussion

This section proposed an original methodology for designing a spiral flight path. This methodology relies on a trade-off between the vertical resolution and the area of constant illumination. The trade-off was successfully validated using actual SAR data from a flight campaign with two different flight paths: conical and cylindrical spirals.

The flight paths had roughly the same minimum and maximum heights and maximum radius. Originally, though, they were supposed to have the same mean radius. However, the height variation had to be shortened for practical reasons. So, the flight paths ended up with the same maximum radius. Also, the trade-off curve in Figure 4.31 had to be redrawn to compare theory and practice better. Using the mean radius would be preferable for future designs since all fight paths in the trade-off curve would have the same ground resolution. Furthermore, as pointed out in Subsection 4.4.2, the number of design alternatives could be increased if several trade-off curves were drawn, varying the height limits and the mean (or maximum) radius.

The cylindrical flight path represents the state-of-the-art techniques for providing 3D SAR images with high resolution. Despite that, the conical flight path undoubtedly had a better performance in terms of vertical resolution than the cylindrical flight path, 1.76 m against 2.33 m. Thus, the output images obtained from the quad-corner reflector support the claim made in Subsection 4.1.5.

Note that these results were obtained with a drone-borne SAR system operating on the P-band. Ponce *et al.* [3] achieved a 2.0 m vertical resolution at the L-band for a Luneberg lens; their airborne SAR system performed multiple circular flight paths with constant radii. Given the difference in wavelength, obtaining a finer resolution with the P-band is quite an accomplishment. Therefore, this outcome highlights the potential of drone-borne SAR systems for applications that require high-resolution 3D SAR images.

The quad-corner reflector used in the flight campaign was the first 360° reflector designed by our research group. Nonetheless, given that its resulting 3D image had the artifacts depicted in Figure 4.37, another 360° reflector has been recently designed. Therefore, future works shall benefit from a reflector with a better 3D response that might not require equalization.

For the case study of Section 3.4, the data were also processed using a DTM. Another member of our research group had developed that DTM as part of their research project. However, the methodology was much more complex, and several 2D layers were processed using the BP algorithm. That is why a detailed description of that DTM was not pertinent to this thesis.

Notwithstanding, the FFBP algorithm is a powerful tool for creating a DTM since a 3D SAR image is required. Take, for example, the 3D image in Subsection 4.4.4. It took 49 min 35 s to process with the FFBP algorithm, but if the BP algorithm had been used instead, the processing time would have been more than 20 h. This estimation comes from multiplying the number of voxels by the number of radar positions and comparing the products. As a result, the product of Subsection 4.4.4 is 3.4 times greater than that of the case study in Section 3.4, which took just over seven hours to process (see Table 3.14).

The BP algorithm explained in Section 2.4 has an optional step for determining the synthetic aperture of each pixel or voxel, which can be performed by image block when necessary. However, the proposed FFBP algorithm, detailed in Section 3.2, does not contain such a step. That is why the images lose focus outside the radius of constant illumination (see Figure 4.36). Therefore, if that step is incorporated in future versions of the FFBP algorithm, the quality of the images shall improve, especially on the borders. Even so, the trade-off proposed here could still apply since the resolution would deteriorate outside the area constantly illuminated by the radar.

5 CONCLUSIONS*

This thesis offers two solutions for producing 3D SAR images with a drone-borne SAR system: the FFBP algorithm and conical spiral flight paths. The FFBP algorithm can readily process 2D and 3D images thanks to a flexible tree structure named the modified Morton-order curve. Since it is a space-filling curve, volumetric images are processed in full-3D rather than in several 2D layers. Furthermore, the FFBP algorithm features an original method for generating sub-apertures based on a data mapping approach. As demonstrated in Section 3.3, it works for random flight paths and is faster than recursively calculating weighted mean positions.

The FFBP algorithm produces nearly identical images to those processed with the BP algorithm, only faster. The speed-up factor is up to 21 times for the 3D images and 13 times for 2D images, with a phase error standard deviation of ~12°. For higher image quality, with a phase error standard deviation of ~4°, the speed-up factor is 10 and 6 times for the 3D and 2D images, respectively.

This thesis also provides a statistical phase error analysis to determine how the FFBP setup affects the quality of the output images. In Section 3.4, the same raw radar data were processed with the FFBP algorithm with different parameters to produce several 2D and 3D SAR images. The analysis validates the hypothesis that geometric parameters defined at the beginning of processing can predict the phase error standard deviation at the output. In future works, the linear regression models generated in the analysis could be applied to determine the processing setup from a requirement in phase error.

Since this thesis worked exclusively with P-band data, this statistical analysis should be repeated for other frequencies in the future, as the phase error also depends on the radar wavelength. Furthermore, it should be repeated for different SAR data to create a robust model. Then, this model would help set the processing parameters from a requirement in phase error, making the FFBP algorithm more user-friendly. Further improvements in the FFBP

* The portion of this chapter that addresses the FFBP algorithm contains revised material from two publications by the author: [1] © 2020 IEEE and [2] licensed under CC BY 4.0.

algorithm could include making the range sampling strategy more efficient and handling areas not constantly illuminated by the radar.

The second solution for 3D SAR imaging is conical spiral flight paths. The claim that they can perform better than state-of-the-art solutions is validated with both simulated and actual SAR data. Under similar conditions, the configuration with the best vertical resolution is the spiral flight path with a tomographic aperture perpendicular to the LOS direction. However, as the tilt angle diverges from 90° (i.e., the cylindrical case), the area of constant illumination decreases. Therefore, this thesis proposes a method for designing spiral flight paths that consider the trade-off between area coverage and vertical resolution. The vertical resolution is calculated with a theoretical expression proposed by this thesis, which originates from the reasoning of Ponce *et al.* [3], but with a revised definition for the vertical wavenumber shift.

The simulation results of Section 4.3 suggest that the radar positions closer to the target have more influence over the ground and vertical resolutions. This perception originated a new set of equations, accounting for the distance from the target to the spiral axis. That is an exciting development, given that the original Circular SAR expressions by Ishimaru *et al.* [4] could only be calculated analytically for a target at the axis. However, the near-range influence could not be observed in the flight campaign of Section 4.4, probably due to equalizing the input SAR data on account of the response of the quad-corner reflector. Nonetheless, the results confirm that the conical flight path performs better than the state-of-the-art cylindrical flight path. The P-band vertical resolution of the quad-corner reflector is 1.76 m for the conical flight path and 2.33 m for the cylindrical flight path. In addition, the proposed theoretical expression predicts the ratio between the two vertical resolutions with a 1.5 % accuracy.

Section 4.3 also demonstrated that Spiral SAR converges faster than Multi-Circular SAR as the number of turns or tracks increases. However, the hypothesis that decorrelation does not affect Spiral SAR could neither be confirmed nor denied. The reason is that this thesis only presented P-band data results; thus, once again, the analysis should be repeated for other frequency bands.

Overall, the two solutions proposed by this thesis are essential milestones for enabling subsurface tomography. Possible applications include, for instance: detecting ant nests in the soil with forest or vegetation cover, mapping the biomass of tubers, surveying underground galleries in a mining site, and estimating soil properties. Furthermore, since the conical flight paths produce images with high resolutions in the vertical direction, they have great potential to enable the estimation of soil properties in different soil horizons. Indeed, the P-band vertical resolution achieved in this thesis (1.76 m) is better than that reported by Ponce *et al.* [3] for the L-band (2.0 m), also making evident the advantage of working with a drone-borne SAR system. Finally, the FFBP algorithm already proved its value by facilitating the analyses of Chapter 4.

BIBLIOGRAPHY

[1] M. I. Skolnik, *Radar Handbook, Third Edition*. McGraw-Hill Education, 2008.

[2] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, “A tutorial on synthetic aperture radar,” *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 1, pp. 6–43, Mar. 2013, DOI: 10.1109/MGRS.2013.2248301.

[3] I. Woodhouse, *Introduction to Microwave Remote Sensing*, 1ª edição. Speckled Press, 2015.

[4] L. J. Cutrona, W. E. Vivian, E. N. Leith, and G. O. Hall, “A High-Resolution Radar Combat-Surveillance System,” *IRE Trans. Mil. Electron.*, vol. MIL–5, no. 2, pp. 127–131, Apr. 1961, DOI: 10.1109/IRET-MIL.1961.5008330.

[5] A. Flores, K. Herndon, R. Thapa, and E. Cherrington, *The SAR Handbook: Comprehensive Methodologies for Forest Monitoring and Biomass Estimation*. 2019. DOI: 10.25966/nr2c-s697.

[6] “Synthetic Aperture Radar (SAR) Satellites | Instrumentation | UNAVCO.” https://www.unavco.org/instrumentation/geophysical/imaging/sar-satellites/sar-satellites.html (accessed Jun. 17, 2022).

[7] R. Acevo-Herrera, A. Aguasca, J. J. Mallorqui, and X. Fabregas, “High-compacted FM-CW SAR for boarding on small UAVs,” in *2009 IEEE International Geoscience and Remote Sensing Symposium*, Jul. 2009, vol. 2, p. II-543-II–546. DOI: 10.1109/IGARSS.2009.5418139.

[8] L. Moreira *et al.*, “A Drone-borne Multiband DInSAR: Results and Applications,” in *2019 IEEE Radar Conference (RadarConf)*, Apr. 2019, pp. 1–6. DOI: 10.1109/RADAR.2019.8835653.

[9] L. Moreira *et al.*, “Drone-Borne P-band Single-Pass InSAR,” in *2020 IEEE Radar Conference (RadarConf20)*, Sep. 2020, pp. 1–6. DOI: 10.1109/RadarConf2043947.2020.9266502.

[10] G. Oré *et al.*, “Predicting Sugarcane Harvest Date and Productivity with a Drone-Borne Tri-Band SAR,” *Remote Sens.*, vol. 14, no. 7, Art. no. 7, Jan. 2022, DOI: 10.3390/rs14071734.

[11] G. Oré *et al.*, “Crop Growth Monitoring with Drone-Borne DInSAR,” *Remote Sens.*, vol. 12, no. 4, Art. no. 4, Jan. 2020, DOI: 10.3390/rs12040615.

[12] D. Luebeck *et al.*, “Drone-borne Differential SAR Interferometry,” *Remote Sens.*, vol. 12, no. 5, Art. no. 5, Jan. 2020, DOI: 10.3390/rs12050778.

[13] J. A. Góes, V. Castro, L. Sant’Anna Bins, and H. E. Hernandez-Figueroa, “3D Fast Factorized Back-Projection in Cartesian Coordinates,” in *2020 IEEE Radar Conference (RadarConf20)*, Sep. 2020, pp. 1–6. DOI: 10.1109/RadarConf2043947.2020.9266484.

[14] J. A. Góes, V. Castro, L. Sant’Anna Bins, and H. E. Hernandez-Figueroa, “Spiral SAR Imaging with Fast Factorized Back-Projection: A Phase Error Analysis,” *Sensors*, vol. 21, no. 15, Art. no. 15, Jan. 2021, DOI: 10.3390/s21155099.

[15] J. A. Góes, *3D FFBP*. Zenodo, 2021. DOI: 10.5281/zenodo.4883259.

[16] *IEEE Standard Letter Designations for Radar-Frequency Bands*. 2020. [Online]. Available: https://doi.org/10.1109/IEEESTD.2020.8999849

[17] O. Frey, C. Magnard, M. Ruegg, and E. Meier, "Focusing of Airborne Synthetic Aperture Radar Data From Highly Nonlinear Flight Tracks," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 6, pp. 1844–1858, Jun. 2009, DOI: 10.1109/TGRS.2008.2007591.

[18] C. Cafforio, C. Prati, and F. Rocca, "SAR data focusing using seismic migration techniques," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, no. 2, pp. 194–207, Mar. 1991, DOI: 10.1109/7.78293.

[19] C. Wu, K. y. Liu, and M. Jin, "Modeling and a Correlation Algorithm for Spaceborne SAR Signals," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-18, no. 5, pp. 563–575, Sep. 1982, DOI: 10.1109/TAES.1982.309269.

[20] M. Y. Jin and C. Wu, "A SAR correlation algorithm which accommodates large-range migration," *IEEE Trans. Geosci. Remote Sens.*, vol. GE-22, no. 6, pp. 592–597, Nov. 1984, DOI: 10.1109/TGRS.1984.6499176.

[21] R. K. Raney, H. Runge, R. Bamler, I. G. Cumming, and F. H. Wong, "Precision SAR processing using chirp scaling," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 786–799, Jul. 1994, DOI: 10.1109/36.298008.

[22] A. Moreira, J. Mittermayer, and R. Scheiber, "Extended chirp scaling algorithm for air- and spaceborne SAR data processing in stripmap and ScanSAR imaging modes," *IEEE Trans. Geosci. Remote Sens.*, vol. 34, no. 5, pp. 1123–1136, Sep. 1996, DOI: 10.1109/36.536528.

[23] M. Duersch, "Backprojection for Synthetic Aperture Radar," *Theses Diss.*, Jun. 2013, [Online]. Available: https://scholarsarchive.byu.edu/etd/4060

[24] C. Werner, T. Strozzi, U. Wegmuller, and A. Wiesmann, "SAR geocoding and multi-sensor image registration," in *IEEE International Geoscience and Remote Sensing Symposium*, Jun. 2002, vol. 2, pp. 902–904 vol.2. DOI: 10.1109/IGARSS.2002.1025723.

[25] D. C. Munson, J. D. O'Brien, and W. K. Jenkins, "A tomographic formulation of spotlight-mode synthetic aperture radar," *Proc. IEEE*, vol. 71, no. 8, pp. 917–925, Aug. 1983, DOI: 10.1109/PROC.1983.12698.

[26] O. Ponce, P. Prats-Iraola, R. Scheiber, A. Reigber, and A. Moreira, "First Airborne Demonstration of Holographic SAR Tomography With Fully Polarimetric Multicircular Acquisitions at L-Band," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6170–6196, Oct. 2016, DOI: 10.1109/TGRS.2016.2582959.

[27] A. Doerry, E. Bishop, and J. Miller, "Basics of Backprojection Algorithm for Processing Synthetic Aperture Radar Images," Sandia National Laboratories, US, SAND2016-1682, Feb. 2016.

[28] Earth Resources Observation And Science (EROS) Center, "Shuttle Radar Topography Mission (SRTM) 1 Arc-Second Global." U.S. Geological Survey, 2017. DOI: 10.5066/F7PR7TFT.

[29] A. W. Doerry, E. Bishop, J. Miller, V. Horndt, and D. Small, "Designing interpolation kernels for SAR data resampling," in *Radar Sensor Technology XVI*, May 2012, vol. 8361, pp. 460–476. DOI: 10.1117/12.918106.

[30] J. W. McCorkle and M. Rofheart, “Order N^2 log(N) backprojector algorithm for focusing wide-angle wide-bandwidth arbitrary-motion synthetic aperture radar,” in *Radar Sensor Technology*, Jun. 1996, vol. 2747, pp. 25–36. DOI: 10.1117/12.243085.

[31] L. M. H. Ulander, H. Hellsten, and G. Stenstrom, “Synthetic-aperture radar processing using fast factorized back-projection,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 3, pp. 760–776, Jul. 2003, DOI: 10.1109/TAES.2003.1238734.

[32] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Pearson New International Edition PDF eBook*. Pearson Education, 2013.

[33] Q. Dong, G.-C. Sun, Z. Yang, L. Guo, and M. Xing, “Cartesian Factorized Backprojection Algorithm for High-Resolution Spotlight SAR Imaging,” *IEEE Sens. J.*, vol. 18, no. 3, pp. 1160–1168, Feb. 2018, DOI: 10.1109/JSEN.2017.2780164.

[34] H. Sun, Z. Sun, T. Chen, Y. Miao, J. Wu, and J. Yang, “An Efficient Backprojection Algorithm Based on Wavenumber-Domain Spectral Splicing for Monostatic and Bistatic SAR Configurations,” *Remote Sens.*, vol. 14, no. 8, Art. no. 8, Jan. 2022, DOI: 10.3390/rs14081885.

[35] O. Ponce *et al.*, “Fully Polarimetric High-Resolution 3-D Imaging With Circular SAR at L-Band,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 6, pp. 3074–3090, Jun. 2014, DOI: 10.1109/TGRS.2013.2269194.

[36] X. Song and W. Yu, “Processing video-SAR data with the fast backprojection method,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 6, pp. 2838–2848, Dec. 2016, DOI: 10.1109/TAES.2016.150581.

[37] T. M. Marston, D. S. Plotnick, and K. L. Williams, “Three dimensional fast factorized back projection for sub-sediment imaging sonars,” in *OCEANS 2019 MTS/IEEE SEATTLE*, Oct. 2019, pp. 1–6. DOI: 10.23919/OCEANS40490.2019.8962594.

[38] A. F. Yegulalp, “Fast backprojection algorithm for synthetic aperture radar,” in *Proceedings of the 1999 IEEE Radar Conference. Radar into the Next Millennium (Cat. No.99CH36249)*, Apr. 1999, pp. 60–65. DOI: 10.1109/NRC.1999.767270.

[39] H. Zhang, J. Tang, R. Wang, Y. Deng, W. Wang, and N. Li, “An Accelerated Backprojection Algorithm for Monostatic and Bistatic SAR Processing,” *Remote Sens.*, vol. 10, no. 1, Art. no. 1, Jan. 2018, DOI: 10.3390/rs10010140.

[40] Morton, G.M., “A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing,” IBM, Otawa, ON, Canada, 1966. Accessed: May 03, 2022. [Online]. Available: https://dominoweb.draco.res.ibm.com/reports/Morton1966.pdf

[41] M. Connor and P. Kumar, “Fast construction of k-nearest neighbor graphs for point clouds,” *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 4, pp. 599–608, Jul. 2010, DOI: 10.1109/TVCG.2010.9.

[42] E. J. Hughes and M. Leyland, “Using multiple genetic algorithms to generate radar point-scatterer models,” *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 147–163, Jul. 2000, DOI: 10.1109/4235.850655.

[43] A. Ishimaru, Tsz-King Chan, and Y. Kuga, “An imaging technique using confocal circular synthetic aperture radar,” *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 5, pp. 1524–1530, Sep. 1998, DOI: 10.1109/36.718856.

[44] L. Moore, L. Potter, and J. Ash, “Three-dimensional position accuracy in circular synthetic aperture radar,” *IEEE Aerosp. Electron. Syst. Mag.*, vol. 29, no. 1, pp. 29–40, Jan. 2014, DOI: 10.1109/MAES.2014.130076.

[45] D. Just and R. Bamler, “Phase statistics of interferograms with applications to synthetic aperture radar,” *Appl. Opt.*, vol. 33, no. 20, pp. 4361–4368, Jul. 1994, DOI: 10.1364/AO.33.004361.

[46] R. Bamler and P. Hartl, “Synthetic Aperture Radar Interferometry.,” *Inverse Probl.*, vol. 14, Jan. 1998.

[47] A. Moreira *et al.*, “Multi-baseline spaceborne SAR imaging,” in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Jul. 2016, pp. 1420–1423. DOI: 10.1109/IGARSS.2016.7729363.

[48] A. Reigber and A. Moreira, “First demonstration of airborne SAR tomography using multibaseline L-band data,” *IEEE Trans. Geosci. Remote Sens.*, vol. 38, no. 5, pp. 2142–2152, Sep. 2000, DOI: 10.1109/36.868873.

[49] O. Frey and E. Meier, “3-D Time-Domain SAR Imaging of a Forest Using Airborne Multibaseline Data at L- and P-Bands,” *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 10, pp. 3660–3664, Oct. 2011, DOI: 10.1109/TGRS.2011.2128875.

[50] X. X. Zhu and R. Bamler, “Very High Resolution Spaceborne SAR Tomography in Urban Environment,” *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 12, pp. 4296–4308, Dec. 2010, DOI: 10.1109/TGRS.2010.2050487.

[51] O. Ponce, H. Joerg, R. Scheiber, P. Prats, I. Hajnsek, and A. Reigber, “First study on holographic SAR tomography over agricultural crops at C-/X-band,” in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Jul. 2016, pp. 7403–7406. DOI: 10.1109/IGARSS.2016.7730931.

[52] O. Ponce, R. Scheiber, P. Prats, I. Hajnsek, and A. Reigber, “Multi-dimensional airborne holographic SAR tomography reconstruction for glaciers at L-/P-band,” in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Jul. 2016, pp. 9–12. DOI: 10.1109/IGARSS.2016.7728993.

[53] C. Lv, B. Deng, Y. Zhang, Q. Yang, and H. Wang, “Terahertz Spiral SAR Imaging Algorithms and Simulations,” in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Chongqing, China, Jun. 2020, pp. 1746–1750. DOI: 10.1109/ITNEC48623.2020.9085036.

[54] D. J. Sego, H. Griffiths, and M. C. Wicks, “Waveform and aperture design for low-frequency RF tomography,” *IET Radar Sonar Amp Navig.*, vol. 5, no. 6, pp. 686–696, Jul. 2011, DOI: 10.1049/iet-rsn.2011.0014.

[55] D. J. Sego and H. Griffiths, “Tomography using digital broadcast television — Flight test and interim results,” in *2016 IEEE Radar Conference (RadarConf)*, May 2016, pp. 1–6. DOI: 10.1109/RADAR.2016.7485165.

[56] F. Gatelli, A. Monti Guamieri, F. Parizzi, P. Pasquali, C. Prati, and F. Rocca, “The wavenumber shift in SAR interferometry,” *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 855–865, Jul. 1994, DOI: 10.1109/36.298013.

[57] S. Guillaso, A. Reigber, L. Ferro-Famil, and E. Pottier, “Range resolution improvement of airborne SAR images,” *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 1, pp. 135–139, Jan. 2006, DOI: 10.1109/LGRS.2005.859943.

[58] *RBAC-E 94 EMD 01 – Requisitos gerais para aeronaves não tripuladas de uso civil*. Accessed: Mar. 06, 2022. [Online]. Available: https://www.anac.gov.br/assuntos/legislacao/legislacao-1/rbha-e-rbac/rbac/rbac-e-94/

[59] *14 CFR Part 107 – Small Unmanned Aircraft Systems*. Accessed: Mar. 06, 2022. [Online]. Available: https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-107

[60] *Commission Implementing Regulation (EU) 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft (Text with EEA relevance)*. Accessed: Mar. 06, 2022. [Online]. Available: https://www.anac.gov.br/assuntos/legislacao/legislacao-1/rbha-e-rbac/rbac/rbac-e-94/@@display-file/arquivo_norma/RBACE94EMD01.pdf

[61] “Matrice 600 Pro - Informações sobre o produto - DJI,” *DJI Official*. https://www.dji.com/br/matrice600-pro/info (accessed Apr. 07, 2022).

[62] A. W. Doerry, “Reflectors for SAR performance testing-second edition,” Sandia National Laboratories, US, SAND2014-0882, Feb. 2014. DOI: 10.2172/1204079.

[63] “Echomaster Radar Reflector: Features, Installation, Design Principle.” Davis Instruments Corporation, Jan. 2009. Accessed: Apr. 11, 2022. [Online]. Available: https://cdn.shopify.com/s/files/1/0515/5992/3873/files/00155-010_IM_00152_153_157.pdf

[64] “Compatible Array Sizes for Basic Operations - MATLAB & Simulink.” https://www.mathworks.com/help/matlab/matlab_prog/compatible-array-sizes-for-basic-operations.html (accessed Jun. 15, 2022).

APPENDIX – FFBP CODE

Section 3.2 presented the FFBP algorithm, the latest version of which is openly available on Zenodo [15]. This appendix explains some blocks of code, drawing a parallel with the equations presented in Section 3.2. Those equations shall be repeated here for convenience. Section A.1 refers to the line of codes that implement the FFBP algorithm. Section A.2 explains the function that creates the splitting scheme.

A.1 IMPLEMENTING THE ALGORITHM

A.1.1 Preparation Step

This subsection explains a few operations executed before the code divides the image into blocks to be processed in parallel.

```
% Adjustment of the number of azimuth lines --------------------
Nazm=Lc^steps*ceil(Nazm0/Lc^steps);
rootXYZ = [rootXYZ; repmat(rootXYZ(end,:),Nazm-Nazm0,1)];
rootData = [rootData; zeros(Nazm-Nazm0,Nrng)];
% ---------------------------------------------------------------
```

The first block of code shows the chosen solution for the following issue: how do we process data from an arbitrary number of radar root positions $\widetilde{K}_0$ (`Nazm0`)? The solution here is to find the first integer K_0 (`Nazm`) that:

$$K_0 > \widetilde{K}_0 \tag{A.1}$$

$$K_0 \propto L^N \tag{A.2}$$

where L (`Lc`) is the number of subapertures combined at each recursion step, and N is the number of steps. Then, the last radar root position in $\boldsymbol{r}_0$ (`rootXYZ`) is repeated $K_0 - \widetilde{K}_0$ times and the SAR root data s_0 is padded with zeros.

```
% Set of all sub-aperture phase centers ------------------------
allX=spline(1:Nazm,rootXYZ(:,1),1:0.5:Nazm)';
allY=spline(1:Nazm,rootXYZ(:,2),1:0.5:Nazm)';
allZ=spline(1:Nazm,rootXYZ(:,3),1:0.5:Nazm)';
allXYZ=[allX allY allZ];
clear allX allY allZ
% ---------------------------------------------------------------
```

Let $\boldsymbol{\Omega}_0$ (`allXYZ`) be the set that comprises all radar root positions and the midpoints between them. This block of code uses interpolation to calculate $\boldsymbol{\Omega}_0$ as (3.11):

$$\boldsymbol{\Omega}_0(i) = \boldsymbol{r}_0\left({}^{i}\!/_{2}\right) \tag{A.3}$$

```
102 % Data upsampling -------------------------------------------------
103 upSampledData=interpft(rootData,upSampling*Nrng,2);
104 rngIncr=rngBin/upSampling;
105 clear rootData
106 % ----------------------------------------------------------------
```

Here, the code upsamples the SAR root data. Also, the code calculates the updated range sampling interval δ_{sr} (`rngIncr`), which will be used throughout the code.

The goal of upsampling is to avoid aliasing when interpolating the data. According to Doerry *et al.* [29], linear interpolation requires a fractional bandwidth of up to 0.184 to avoid aliasing. Therefore, if the root SAR data had a fractional bandwidth of 1.0, the upsampling rate (`upSampling`) should be at least 1.0/0.184 = 5.43.

The upsampling rate is an input provided by the user. Since the function `interpft` expects an integer at its second argument (`upSampling*Nrng`), the user must choose an integer value for the upsampling rate.

```
116 % Creating cell arrays -------------------------------------------
117 Dx = firstSplit(1);
118 Dy = firstSplit(2);
119 Dz = firstSplit(3);
120 cellSize = [Dy Dx Dz];
121
122 dataCell = cell(cellSize);
123 xCell = cell(cellSize);
124 yCell = cell(cellSize);
125 zCell = cell(cellSize);
126 % ----------------------------------------------------------------
```

The imaged area or volume is split evenly into image blocks to be processed in parallel with the FFBP algorithm. The data are managed by creating a cell array for each output matrix, i.e., processed SAR data and voxel coordinates. All cell arrays have the same number of elements, and each cell index is associated with an image block. When an image block is processed, its results are stored in the corresponding data cell. Then, after processing all image blocks, each cell array is converted into a matrix that combines data for the whole output image (lines 409-433).

```
% Arranging the area/volume blocks in the cells ----------------
cellDim=newDim./firstSplit;

x1 = (-(Dx-1)/2:(Dx-1)/2);
y1 = (-(Dy-1)/2:(Dy-1)/2);
z1 = (-(firstSplit(3)-1)/2:(Dz-1)/2);

x2 = repmat(x1,Dy,Dz(3));
y2 = repmat(y1',Dx*Dz,1);
z2 = repmat(z1,Dx*Dz,1);

cellCntr = [x2(:) y2(:) z2(:)].*cellDim + rootCntr;
cellCount = size(cellCntr,1);
% -------------------------------------------------------------
```

The modified Morton order curve is used here to determine the location of each image block and arrange them within the cell arrays.

A.1.2 Parallel Processing

This subsection explains some operations that occur inside the parallel for loop. Except in 0, they also occur inside the for loop that controls the recursion steps.

```
% Split scheme -------------------------------------------------
Dpx = partition(s,1);
Dpy = partition(s,2);
Dpz = partition(s,3);
Dp = Dpx*Dpy*Dpz;
childCount = Dp*parentCount;
childDim = parentDim./[Dpx Dpy Dpz];
% -------------------------------------------------------------
```

In this block of code, the number of divisions in each direction (D_x, D_y, and D_z) are retrieved from the splitting scheme. As the number of child subimages increases by a factor $D_n = D_x D_y D_z$, their dimensions decrease according to (3.13-3.15):

$$A_{x_n} = \frac{A_{x_{n-1}}}{D_x}, \quad A_{y_n} = \frac{A_{y_{n-1}}}{D_y}, \quad A_{z_n} = \frac{A_{z_{n-1}}}{D_z} \tag{A.4}$$

where A_{x_n}, A_{y_n} and A_{z_n} are the dimensions of the child subimage (`childDim`) and $A_{x_{n-1}}$, $A_{y_{n-1}}$ and $A_{z_{n-1}}$ are the dimensions of the parent subimage (`parentDim`).

```
% (x,y,z) coordinates ------------------------------------------
x1 = (-(Dpx-1)/2:(Dpx-1)/2);
y1 = (-(Dpy-1)/2:(Dpy-1)/2);
z1 = (-(Dpz-1)/2:(Dpz-1)/2);

x2 = repmat(x1',Dpy*Dpz,1);
y2 = repmat(y1,Dpx,Dpz);
z2 = repmat(z1,Dpx*Dpy,1);

auxCntr = [x2(:) y2(:) z2(:)].*childDim;

childCntr=zeros(childCount,3);
for p=1:parentCount
    c = (p-1)*Dp+(1:Dp);
    childCntr(c,:)=auxCntr+parentCntr(p,:);
end
height=interp2(xDEM,yDEM,zDEM,childCntr(:,1),childCntr(:,2),...
    'spline');
% --------------------------------------------------------------
```

This block of code finds the position of each child subimage center in $\boldsymbol{h}_n$ (`childCntr`) by creating a building block of the modified Morton order curve around the origin, then placing this building block at each parent subimage center in $\boldsymbol{h}_{n-1}$ (`parentCntr`). In order to create the building block, the code first finds all possible values of its coordinates (`x1`, `y1`, `z1`), which are dimensionless at first.

Then, the code repeats these values in a certain way (`x2`, `y2`, `z2`), rearranging them to create a pattern similar to a truth table. Figure A.1 shows this process for $D_x = D_y = 3$ and $D_z = 2$. When the table is multiplied by the dimensions of the child subimage, the result is the coordinates $\tilde{x}$, $\tilde{y}$, and $\tilde{z}$ (`auxCntr`) of a Z-shaped curve.

The corresponding equations are (3.13-3.17):

$$x(d_x) = A_{x_n}\left(d_x - \frac{(D_x - 1)}{2}\right), \quad d_x = 0, 1, \dots, D_x - 1 \tag{A.5}$$

$$y(d_y) = A_{y_n}\left(d_y - \frac{(D_y - 1)}{2}\right), \quad d_y = 0, 1, \dots, D_y - 1 \tag{A.6}$$

$$z(d_z) = A_{z_n}\left(d_z - \frac{(D_z - 1)}{2}\right), \quad d_z = 0, 1, \dots, D_z - 1 \tag{A.7}$$

$$\boldsymbol{h}_n(c) = [\tilde{x}(d) \quad \tilde{y}(d) \quad \tilde{z}(d)] + \boldsymbol{h}_{n-1}(p) \tag{A.8}$$

$$c = pD_n + d \tag{A.9}$$

where $d = 0, 1, \ldots, D_n - 1$, p refers to a parent subimage, c refers to a child subimage, and x, y and z are the possible coordinates for the building block in meters.

x1

-1	0	1

y1

-1	0	1

z1

-0.5	0.5

x2

-1
0
1
-1
0
1
-1
0
1
-1
0
1
-1
0
1
-1
0
1

y2

-1	0	1	-1	0	1
-1	0	1	-1	0	1
-1	0	1	-1	0	1

z2

-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5
-0.5	0.5

[x2(:) y2(:) z2(:)]

-1	-1	-0.5
0	-1	-0.5
1	-1	-0.5
-1	0	-0.5
0	0	-0.5
1	0	-0.5
-1	1	-0.5
0	1	-0.5
1	1	-0.5
-1	-1	0.5
0	-1	0.5
1	-1	0.5
-1	0	0.5
0	0	0.5
1	0	0.5
-1	1	0.5
0	1	0.5
1	1	0.5

Figure A.1 – Creating a pattern similar to a truth table for $D_x = D_y = 3$ and $D_z = 2$.

Finally, the code uses the DEM and $\boldsymbol{h}_n$ to interpolate the terrain height H_{DEM} (line 227). Recall that $\boldsymbol{h}_{n-1}$ and $\boldsymbol{h}_n$ have no topographic information.

```
230 % Mapping sequences ------------------------------------------
231 n1 = repmat(1:length(parentSx),Dpx,1);
232 n2 = repmat(1:length(parentSy),Dpy,1);
233 n3 = repmat(1:length(parentSz),Dpz,1);
234
235 m1 = repmat((0:(Dpx-1))',1,length(parentSx));
236 m2 = repmat((0:(Dpy-1))'*Dpx,1,length(parentSy));
237 m3 = repmat((0:(Dpz-1))'*Dpx*Dpy,1,length(parentSz));
238
239 childSx = parentSx(n1(:))*Dp+m1(:);
240 childSy = parentSy(n2(:))*Dp+m2(:);
241 childSz = parentSz(n3(:))*Dp+m3(:);
242 % ------------------------------------------------------------
```

This block of code calculates the recurrent sequences that will be used later to map the 1D data into a 2D/3D matrix. The corresponding equations are (3.29-3.31):

$$q_{x_n}(uD_x + d_x) = D_n q_{x_{n-1}}(u) + d_x \tag{A.10}$$

$$q_{y_n}\left(vD_y + d_y\right) = D_n q_{y_{n-1}}(v) + d_y D_x \tag{A.11}$$

$$q_{z_n}(wD_z + d_z) = D_n q_{z_{n-1}}(w) + d_z D_x D_y \tag{A.12}$$

where q_{x_n}, q_{y_n}, and q_{z_n} are the child sequences (childSx, childSy, childSz) and $q_{x_{n-1}}$, $q_{y_{n-1}}$, and $q_{z_{n-1}}$ are the parent sequences (parentSx, parentSy, parentSz).

```
% Sub-apertures ----------------------------------------------
Nc=round(Np/Lc);
k=1:Nc;
j=zeros(1,1,Lc); j(:)=1:Lc;
Jc=(k-1)'*Lc+j;
childXYZ=allChildrenXYZ(Lc^s:2*Lc^s:end,:);
% ---------------------------------------------------------------
```

Here, the code defines the child subaperture centers in $\boldsymbol{r}_n$ (childXYZ) using the general case (3.12):

$$\boldsymbol{r}_n(k) = \boldsymbol{\Omega}_0\left((2k+1)L^n - 1\right) \tag{A.13}$$

where $k = 0, 1, \ldots, K_n - 1$. Furthermore, the code creates a $K_n \times 1 \times L$ matrix (Jc) that maps which parent subapertures are combined into each child subaperture. The k^{th} row in Jc corresponds to the set:

$$l \in \Lambda_{n,k} = \{kL + b | b = 0, 1, \ldots, L-1\} \tag{A.14}$$

```
% Child sub-aperture <--> parent sub-aperture ------------------
vec2parent=zeros(Nc,3,Lc);
for j=1:Lc
    vec2parent(:,:,j)=parentXYZ(Jc(k,1,j),:)-childXYZ;
end
rng2parent=vecnorm(vec2parent,2,2);
% ---------------------------------------------------------------
```

Let $\overline{CP}_n$ (rng2parent) be the set of distances between each child subaperture and their corresponding parent subapertures in $\boldsymbol{r}_{n-1}$ (parentXYZ). This block of code uses the Euclidean norm to obtain $\overline{CP}_n$, according to the expression (3.19):

$$\overline{CP}_n(k,l) = \|\boldsymbol{r}_{n-1}(l) - \boldsymbol{r}_n(k)\| \tag{A.15}$$

Though both $\boldsymbol{r}_{n-1}$ and $\boldsymbol{r}_n$ have three columns to account for coordinates in space, $\boldsymbol{r}_{n-1}$ has K_{n-1} rows, while $\boldsymbol{r}_n$ has $K_n = K_{n-1}/L$ rows. Thus, their sizes are incompatible for executing basic operations [64]. Therefore, the data are rearranged, making $\overline{CP}_n$ a $K_n \times 1 \times L$ matrix. Figure A.2 illustrates the different matrix dimensions for $K_n = 4$ and $L = 2$.

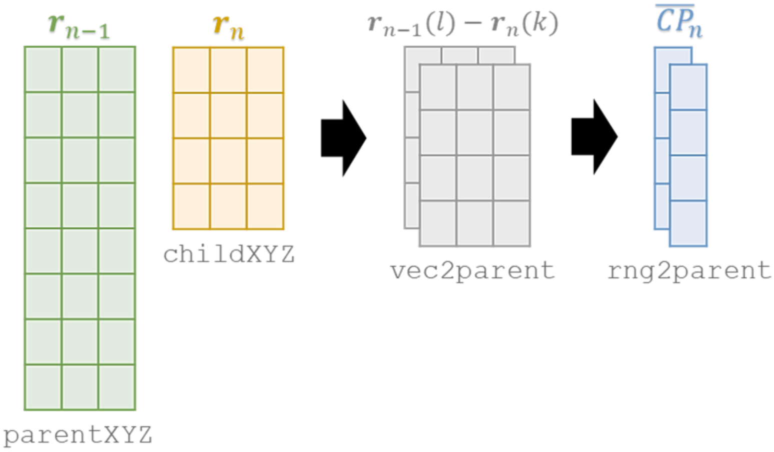


Figure A.2– Matrix dimensions: $\boldsymbol{r}_{n-1}$, $\boldsymbol{r}_n$ and $\overline{CP}_n$.

A.1.2.1 For each child subimage

The following operations occur within a for loop controlled by c.

```
% Child sub-aperture <--> Data sample --------------------------
vec2img=(childCntr(c,:)+[0 0 height(c)]-childXYZ);
rng2img=vecnorm(vec2img,2,2);
child2sample=rng2img+rngIncr*(m-(Mc+1)/2);
% --------------------------------------------------------------
```

Let $\overline{CS}_{n,c}$ (`child2sample`) be the set of distances between each child subaperture and the data samples within the current subimage. This block of code calculates $\overline{CS}_{n,c}$ using both the Euclidean norm and vector scaling. The corresponding equations are (3.18, 3.20):

$$\tilde{\boldsymbol{h}}_{n,c} = \boldsymbol{h}_n(c) + [0 \quad 0 \quad H_{DEM}(\boldsymbol{h}_n(c))] \tag{A.16}$$

$$\overline{CS}_{n,c}(k,m) = \left\|\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)\right\| + \delta_{sr}\left(m - \frac{(M_n - 1)}{2}\right) \tag{A.17}$$

where $m = 0, 1, \dots, M_n - 1$, with M_n being the number of range samples at the n^{th} node, δ_{sr} is the range sampling interval, and $\tilde{\boldsymbol{h}}_{n,c}$ is the center of the child subimage, now including topographic information.

No variable that depends on the range samples represents positions in space and vice-versa. Therefore, the same matrix dimension is used to represent both. Figure A.3 illustrates the matrix dimensions in this block of code for $M_n = 5$. Note that $\overline{CS}_{n,c}$ is a $K_n \times M_n$ matrix.

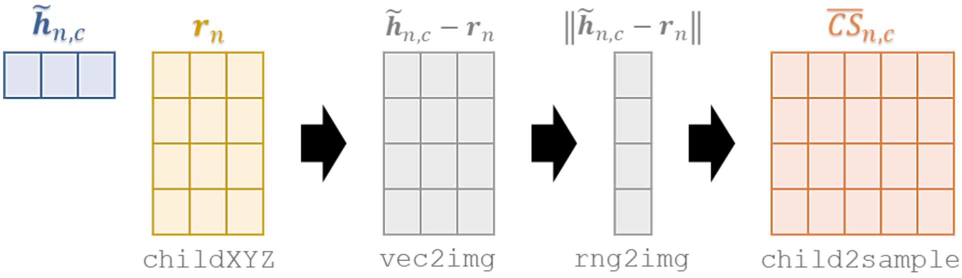


Figure A.3 – Matrix dimensions: $\tilde{\boldsymbol{h}}_{n,c}$, $\boldsymbol{r}_n$ and $\overline{CS}_{n,c}$.

```
308 % Parent sub-aperture <--> Data sample -------------------------
309 cosine=dot(vec2parent./rng2parent,...
310     repmat(vec2img./rng2img,1,1,Lc),2);
311 cosine(rng2parent==0)=0;
312 parent2sample=sqrt(child2sample.^2+rng2parent.^2-...
313     2*rng2parent.*child2sample.*cosine);
314 % --------------------------------------------------------------
```

Let $\overline{PS}_{n,c}$ (`parent2sample`) be the set of distances between each parent subaperture and the data samples within the current child subimage. This block of code calculates $\overline{PS}_{n,c}$ using the law of cosines (3.21, 3.22):

$$cos\left(\theta_{n,c}(k,l)\right) = \frac{\boldsymbol{r}_{n-1}(l) - \boldsymbol{r}_n(k)}{\overline{CP}_n(k,l)} \cdot \frac{\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)}{\left\|\tilde{\boldsymbol{h}}_{n,c} - \boldsymbol{r}_n(k)\right\|} \tag{A.18}$$

$$\overline{PS}_{n,c}(k,l,m) = \sqrt{\overline{CP}_n(k,l)^2 + \overline{CS}_{n,c}(k,m)^2 - 2\overline{CP}_n(k,l)\overline{CS}_{n,c}(k,m)\, cos\left(\theta_{n,c}(k,l)\right)} \tag{A.19}$$

The first term in equation (A.18) depends on k and l, while the second term depends only on k. That translates into two matrices of different sizes. However, the dot product function (lines 309-310) requires that both input matrices have the same size. Therefore, the code makes L copies of the second input matrix. Figure A.4 illustrates this process.

On the other hand, all terms inside the square root in equation (A.19) represent matrices with compatible sizes for basic operations [64]. Therefore, calculating $\overline{PS}_{n,c}$ is straightforward (lines 312-313). Figure A.5 shows the different matrix dimensions. Note that $\overline{PS}_{n,c}$ is $K_n \times M_n \times L$.

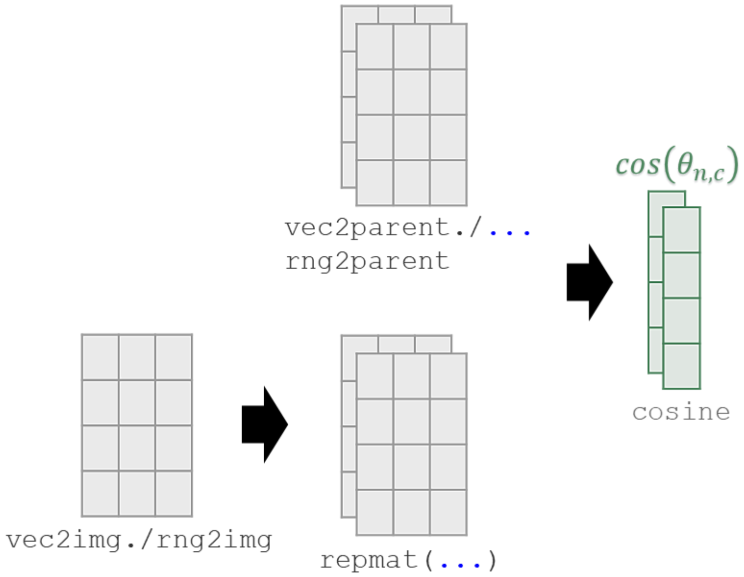


Figure A.4 – Matrix dimensions: $cos(\theta_{n,c})$.

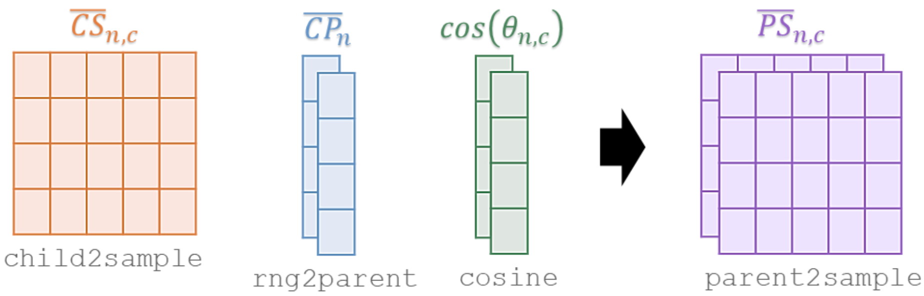


Figure A.5 – Matrix dimensions: $\overline{CS}_{n,c}$, $\overline{CP}_n$, $cos(\theta_{n,c})$ and $\overline{PS}_{n,c}$.

```
% Value of the first range bin ------------------------------------
childR0(:,c)=rng2img-rngIncr*(Mc-1)/2;
rng0=permute(reshape(parentR0(:,p),[Lc,1,Nc]),[3 2 1]);
% ----------------------------------------------------------------
```

Here, the code recalculates the distances from the child subapertures to the first sample in the child data, $\overline{CS}_{n,c}(k,0)$. Then, the data are stored for the next recursion in the c^{th} column of the matrix `childR0`. Also, the code retrieves $\overline{CS}_{n-1,p}(l,0)$ from the last recursion, which had been stored in the p^{th} column of the matrix `parentR0`. Likewise, $\overline{CS}_{n-1,p}(l,0)$ represents the distances from the parent subapertures to the first sample in the parent data. Finally, because $\overline{CS}_{n-1,p}(l,0)$ is a column vector of K_{n-1} elements, the data are rearranged into a $K_n \times 1 \times L$ matrix (`rng0`). The result is similar to the one shown in Figure A.2, but the process is different (see Figure A.6).

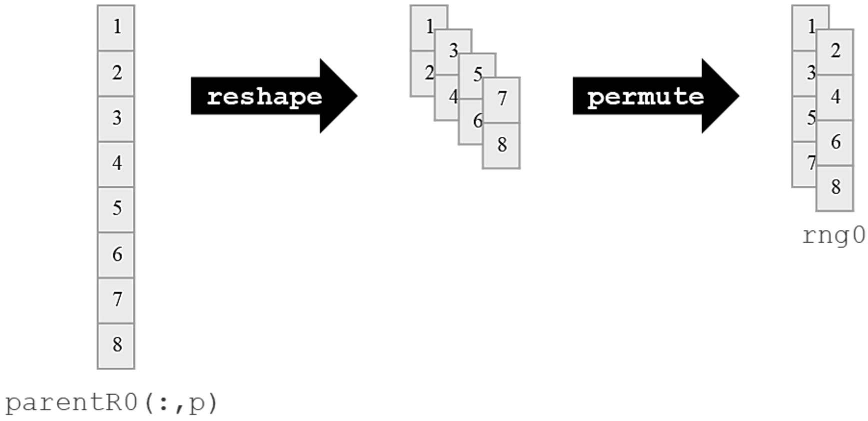


Figure A.6 – Rearranging data with the functions `reshape` and `permute`.

```
% Fracional range index ------------------------------------------
rngIndex=1+(parent2sample-rng0)/rngIncr;
lastIndex=floor(rngIndex);
remainder=rngIndex-lastIndex;
% ------------------------------------------------------------------
```

This block of code calculates what is needed to perform a linear interpolation. The factional indices $\nu_{n,c}$ (`rngIndex`) are given by (3.23):

$$\nu_{n,c}(k,l,m) = \frac{\overline{PS}_{n,c}(k,l,m) - \overline{CS}_{n-1,p}(l,0)}{\delta_{sr}} \tag{A.20}$$

The code also calculates the integer part of $\nu_{n,c}$ (`lastIndex`) and the remainder, which consists of numbers in the interval [0,1). All matrices calculated here are $K_n \times M_n \times L$.

```
336 % Linear interpolation -------------------------------------------
337 i0 = find(rngIndex>1&rngIndex<Mp); % Only valid indices
338 J0 = repmat(Jc,1,Mc,1);
339 P0 = ones(Nc,Mc,Lc).*p;
340 M1 = sub2ind(size(parentData),J0(i0),lastIndex(i0),P0(i0));
341 M2 = sub2ind(size(parentData),J0(i0),lastIndex(i0)+1,P0(i0));
342 interpData=zeros(size(rngIndex));
343 interpData(i0)=parentData(M1).*(1-remainder(i0))+...
344     parentData(M2).*remainder(i0);
345 % ------------------------------------------------------------------
```

This block of code uses linear interpolation to determine the value of $s_{n-1}\big(l,\nu_{n,c}(k,l,m),p\big)$, storing the result in a $K_n \times M_n \times L$ matrix (`interpData`).

The function `sub2ind` (lines 340-341) requires all but the first input to have the same size. Therefore, the matrix that contains all indices l (`Jc`) is repeated M_n times to create a $K_n \times M_n \times L$ matrix (`J0`). In addition, since p is a single value, it is multiplied by a matrix filled with ones to create another matrix of the same size (`P0`).

```
347 % Phase compensation and data combining ------------------------
348 phi=4*pi*(parent2sample-child2sample)/lambda;
349 childData(k,m,c)=sum(interpData.*exp(1i*phi),3)/Lc;
350 % childData(k,m,c)=sum(interpData,3)/Lc;
351 % --------------------------------------------------------------
```

Here, the code calculates the phase compensation term $\varphi_{n,c}$ (`phi`) and performs the coherent sum of the interpolated parent data. The result is the child SAR data associated with the current child subimage, which is stored in the c^{th} panel of s_n (`childData`). The corresponding equations are (3.24, 3.25):

$$\varphi_{n,c}(k,l,m) = \frac{4\pi}{\lambda_0}\left[\overline{PS}_{n,c}(k,l,m) - \overline{CS}_{n,c}(k,m)\right] \tag{A.21}$$

$$s_n(k,m,c) = \sum_{l \in \Lambda_{n,k}} s_{n-1}\big(l, v_{n,c}(k,l,m), p\big)\, e^{j\varphi_{n,c}(k,l,m)} \tag{A.22}$$

where λ_0 is the radar wavelength.

The commented line (350) performs an incoherent sum by removing the phase compensation term (A.21). It was used to generate Figure 3.17(c,f) and Figure 3.41, which demonstrated that the result is completely unsatisfactory without (A.21).

A.1.2.2 Final steps

The following operations occur after the last recursion step ($n = N$).

```
% Coherent sum of remaining data -------------------------------
height=interp2(xDEM,yDEM,zDEM,parentCntr(:,1),...
    parentCntr(:,2),'spline');
serialData=zeros(parentCount,1);
for c=1:parentCount
    rng2img=vecnorm(parentXYZ-
    (parentCntr(c,:)+[0,0,height(c)]),2,2);
    phi=4*pi*rng2img/lambda;
    serialData(c)=serialData(c)+...
        sum(parentData(:,1,c).*exp(1i*phi));
end
% --------------------------------------------------------------
```

This block of code coherently sums the data from the remaining subapertures. The corresponding equations are (3.26, 3.27):

$$s(c) = \sum_k s_N(k, 0, c) e^{j\tilde{\varphi}_c(k)} \tag{A.23}$$

$$\tilde{\varphi}_c(k) = \frac{4\pi}{\lambda_0} \left\| \widetilde{\boldsymbol{h}}_{N,c} - \boldsymbol{r}_N(k) \right\| \tag{A.24}$$

where s (`serialData`) is a 1D matrix that contains the processed SAR data.

```
390 % 1D --> 2D/3D mapping ---------------------------------------
391 [x1,y1,z1]=meshgrid(parentSx,parentSy,parentSz);
392 index=1+x1+y1+z1;
393 dataCell{icell}=serialData(index);
394
395 x2 = parentCntr(1+parentSx,1);
396 y2 = parentCntr(1+parentSy,2);
397 z2 = parentCntr(1+parentSz,3);
398
399 [X3,Y3,Z3] = meshgrid(x2,y2,z2);
400 xCell{icell} = X3;
401 yCell{icell} = Y3;
402 zCell{icell} = Z3;
403 % -------------------------------------------------------------
```

Here, the code maps 1D data into 2D/3D matrices. First (lines 391-392), the code uses the recurrent sequences to map $c \rightarrow (u, v, w)$, according to the expression (3.32):

$$c = q_{x_n}(u) + q_{y_n}(v) + q_{z_n}(w) \tag{A.25}$$

Then (line 393), the code rearranges the processed SAR data into a 2D/3D matrix and stores the result in a cell corresponding to the current image block. Next (lines 395-397), the code uses the recurrent sequences to extract the sets of x, y, and z coordinates from the positions $(\tilde{x}, \tilde{y}, \tilde{z})$ on the modified Morton-order curve (`parentCntr`). It is the inverse process from that illustrated in Figure A.1. Finally (lines 399-402), the x, y, and z coordinates are stored in a mesh grid format in their corresponding cells.

A.2 CREATING THE SPLITTING SCHEME

This section details the latest version of the splitting scheme function, which was used to obtain the experimental results of Section 3.3. The inputs of this function are:

- The dimensions of the output image (`rootDim`);
- The resolution of the output image (`resolution`);
- The first split into image blocks provided by the user (`split0`);
- The number of subapertures that are combined at each recursion L (`Lc`);
- The number of radar root positions $\widetilde{K}_0$ (`Nazm`).

The outputs of the function are:

- The splitting scheme (`splitScheme`);
- The number of recursions steps N (`numSteps`);
- The updated dimensions of the processed image (`newDim`);
- The updated split into image blocks (`firstSplit`);
- The number of pixels/voxels of the output image (`numPoints`).

Recall that the code processes an image larger than the output image, keeping the required resolution unchanged.

```
% Number of pixels/voxels and number of recursion steps --------
numPoints = round(rootDim./resolution);
Mazm = fix(log(Nazm)./log(Lc));
Mxyz = fix(log(numPoints./split0)./log(Lc));
maxMxyz = max(Mxyz);
numSteps = min([Mazm maxMxyz+1]);
% -------------------------------------------------------------
```

Here, the code determines the number of pixels/voxels and the number of recursion steps. The first is calculated directly from the dimensions and resolution of the output image. The second comes from the comparison of two quantities:

- The maximum number of times that the code can combine subapertures in groups of L until reaching $\widetilde{K}_0$;
- The maximum number of divisions from the first split into image blocks to the output number of pixels/voxels if the image is divided by $L \times L \times L$ at each

recursion. This calculation is performed for each direction, but only the maximum value is considered to find the number of recursions.

```
473 % Split scheme for each direction -----------------------------
474 splitScheme = ones(numSteps,3);
475
476
477 for i = 1:3
478     if rootDim(i)~=0
479         if Mxyz(i) <= numSteps-2
480             lastSplit = ceil(numPoints(i)./split0(i)./...
                    Lc.^Mxyz(i));
481             bulk = 2:Mxyz(i)+1;
482             last = Mxyz(i)+2;
483         else
484             lastSplit = floor(numPoints(i)./split0(i)./...
                    Lc.^(Mxyz(i)-1));
485             bulk = 2:Mxyz(i);
486             last = Mxyz(i)+1;
487         end
488         splitScheme(bulk,i) = Lc;
489         splitScheme(last,i) = lastSplit;
490     end
491 end
492 % -------------------------------------------------------------
```

This block of code creates the splitting scheme. The splitting scheme is an $N \times 3$ matrix whose rows contain the number of divisions D_x, D_y, and D_z for each recursion.

The code starts by creating a matrix filled with ones. There are three reasons for this choice. First, suppose the image is 2D (e.g., $A_z = 0$). In that case, the image will not be divided in a certain direction (i.e., $D_z = 1$ for every recursion). Second, because the image is already split into image blocks, $D_x = D_y = D_z = 1$ for the first recursion. Third, suppose one direction has far fewer voxels than the others (lines 479-482). In that case, the required number of voxels for that direction will likely be attained before reaching the last recursion. Thus, the number of divisions in that direction will equal one for the remaining recursions.

Taking that into account, this block of code determines the last recursion (`last`) in which the image will be divided in each direction. The code also calculates the corresponding number of divisions (`lastSplit`), an integer in the interval $[L, L^2)$. Beyond that, after the first recursion and before the last division (`bulk`), the image is divided by L in each direction.

```
494 % Adjustment of the imaged area/volume dimensions ---------------
495 cellPoints = prod(splitScheme,1);
496 firstSplit = ceil(numPoints./cellPoints);
497 newDim = cellPoints.*firstSplit.*resolution;
498 firstSplit(rootDim==0)=1;
499 newDim(rootDim==0)=0;
500 % --------------------------------------------------------------
```

From the splitting scheme, the code calculates the number of pixels/voxels at each image block. Then, the code updates the first split into image blocks and the dimensions of the processed image. Finally (lines 498-499), the code makes the necessary adjustments in the case of a 2D image.