# Speckle Removal and Change Preservation by Distance-Driven Anisotropic Diffusion of Synthetic Aperture Radar Temporal Stacks

A Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

in partial fulfillment

of the requirements for the degree

Master of Science

by

Nazia Tabassum

August 2016

Nazia Tabassum

AUTHOR SIGNATURE

The thesis has been read and approved by the examining committee:

Scott Acton

Advisor

Daniel Weller

Laura Barnes

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, Dean, School of Engineering and Applied Science

August 2016

UNIVERSITY OF VIRGINIA

MASTERS THESIS

# Speckle Removal and Change Preservation by Distance-Driven Anisotropic Diffusion of Synthetic Aperture Radar Temporal Stacks

*Author:*
Nazia TABASSUM

*Supervisor:*
Dr. Scott ACTON

*A thesis submitted in fulfillment of the requirements*
*for the degree of Master of Science*

*in the*

Virginia Image and Video Processing Lab
Department of Electrical Engineering

July 3, 2016

# Declaration of Authorship

I, Nazia TABASSUM, declare that this thesis titled, "Speckle Removal and Change Preservation by Distance-Driven Anisotropic Diffusion of Synthetic Aperture Radar Temporal Stacks" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

*"And above all, watch with glittering eyes the whole world around you because the greatest secrets are always hidden in the most unlikely places. Those who don't believe in magic will never find it."*

Roald Dahl

UNIVERSITY OF VIRGINIA

# *Abstract*

Dr. Scott Acton
Department of Electrical Engineering

Master of Science

**Speckle Removal and Change Preservation by Distance-Driven Anisotropic Diffusion of Synthetic Aperture Radar Temporal Stacks**

by Nazia TABASSUM

Satellite imagery is often collected to perform remote sensing, or uncover changes in our Earth's surface without having to traverse every corner. One application of using radar images is to observe changes in road topography without having to physically examine all of the roads. Unfortunately, radar images are often corrupted by speckle noise, which is a direct result of interference from scatterers at the imaging aperture. There are many speckle-reducing algorithms that work on either the temporal range of the image or the spatial aspect of the image. Traditionally, with synthetic aperture radar images, the mean of time series is utilized to produce a single despeckled image that discards temporal information. We propose a method that smoothes spatially and uses, but also preserves, temporal information. Radar images are often collected of the same region over time. To provide smoothing of such imagery without effacing temporal changes in the scene, we put forth an anisotropic diffusion technique, using a PDE approach. This approach smoothes uniform areas and preserves and enhances edges, such as roads or other features. In order to use this anisotropic diffusion technique, we must first select a homogeneous region in the image to calculate statistics of the speckle noise. This is often done manually, and can be time consuming and inaccurate with a large stack of radar images. We propose a method that uses the temporal information to automatically select a homogeneous region prior to smoothing. Our proposed smoothing method is a statistical approach designed to reduce speckle noise in each image throughout the time series. Results demonstrate the efficacy of the approach on real and synthetic data, showing lower mean squared error than leading methods. The new filter incorporates temporal information essential to detection of potential change events for transportation infrastructure. Change preservation is shown with our approach because temporal information is preserved, while signal-corrupting noise is reduced.

# *Acknowledgements*

First of all, thank you to the United States Department of Transportation for their assistance. This work was supported in part by the U.S. Department of Transportation (US DOT) under Cooperative Agreement #RITARS-14-H-UVA. The views, opinions, findings and conclusions reflected in this presentation are the responsibility of the authors only and do not represent the official policy or position of the US Department of Transportation/Office of the Assistant Secretary for Research and Technology, or any state or other entity.

I would like to thank my advisor, Dr. Scott Acton, for his continuous support and encouragement throughout the duration of my time at the University of Vriginia. You have helped me grow in ways I did not think was possible.

Thank you to all of the members of both VIVA and VITAL labs, whose knowledge, generosity, and curiosity have made the journey incredible. You all are the best!

Thank you to my committee members for giving me their time so that I can share my work with them.

Thank you to my loving family for being with me every step of the way. You know just when to push me, and just how to give me loving warmth when I need it.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **SAR** | **S**ynthetic **A**perture **R**adar |
| **USDOT** | **U**nited **S**tates **D**epartment **O**f **T**ransportation |
| **DOT** | **D**epartment **O**f **T**ransportation |
| **GIS** | **G**eographic **I**nformation **S**ystem |
| **TRE** | **T**ele- **R**ilevamento **E**uropa |
| **SAR** | **S**ynthetic **A**perture **R**adar |
| **SRAD** | **S**peckle **R**educing **A**nisotropic **D**iffusion |
| **DD-SRAD** | **D**istance- **D**riven **S**peckle **R**educing **A**nisotropic **D**iffusion |
| **ENL** | **E**ffective **N**umber of **L**ooks |
| **PDE** | **P**artial **D**ifferential **E**quation |
| **ICOV** | **I**nstantaneous **C**oefficient **O**f **V**ariation |
| **DPAD** | **D**etail **P**reserving **A**nisotropic **D**iffusion |
| **SHP** | **S**tatistically **H**omogeneous **P**ixels |
| **KS** | **K**olmogorov- **S**mirnov |
| **CDF** | **C**umulative **D**istribution **F**unction |
| **PDF** | **P**robability **D**istribution **F**unction |
| **RMSE** | **R**oot **M**ean **S**quare **E**rror |
| **RSS** | **R**oot **S**um of **S**quares |
| **ECDF** | **E**mpirical **C**umulative **D**istribution **F**unction |
| **MSE** | **M**ean **S**quared **E**rror |
| **PSNR** | **P**eak **S**ignal-to-**N**oise **R**atio |
| **Med-SRAD** | **M**edian **S**peckle **R**educing **A**nisotropic **D**iffusion |

# List of Symbols

| | |
|---|---|
| $\triangledown I$ | gradient of image |
| $\|\|$ | magnitude |
| $div$ | divergence |
| $I_0$ | input image |
| $c(x)$ | diffusion coefficient |
| $I_s^t$ | discretely sampled image |
| $s$ | pixel position |
| $\Delta t$ | time step |
| $\overline{\eta_s}$ | window |
| $\mid \overline{\eta_s} \mid$ | size of window |
| $\bar{I}_s$ | average value of window |
| $k_s$ | adaptive filter coefficient |
| $ENL$ | effective number of looks |
| $z'$ | homogeneous region |
| $K$ | damping factor |
| $\Omega$ | 2D coordinate grid |
| $\partial\Omega$ | border of grid |
| $\overrightarrow{n}$ | outer normal to border |
| $q_0(t)$ | speckle scale function |
| $z(t)$ | homogeneous region |
| $P$ | center pixel |
| $d_1(P)$ | pixel value in first image |
| $N$ | number of images in stack |
| $S_N(X)$ | unbiased estimator for pixel CDF |
| $x_i$ | $i^{th}$ element in data vector |
| $c$ | threshold for KS test |
| $\alpha$ | significance level |
| $D_N$ | difference between CDFs |
| $D$ | distance between pixels through time |
| $\sigma^2$ | variance |

*To my loving family and friends: "In this garden of my dreams, spring has blossomed because of you. The colors of these blossoms are mine, yet the fragrance comes from you."*

# Chapter 1

# Introduction

## 1.1 SAR Images

Synthetic aperture radar (SAR) is a form of radar used for imagery. The name comes from the formation of a synthetic aperture by moving a radar antenna over a region of interest, as opposed to using a beam-scanning radar system from a fixed position. The antenna is affixed to a moving device in the air: in this case, a satellite. The distance the satellite moves before returning to its original position creates the synthetic aperture, or synthetic antenna size. As the aperture created does not depend on the length of the physical antenna, we can achieve high imaging resolution with smaller antennas, due to the large distance covered by the satellite (*Radar Basics*).

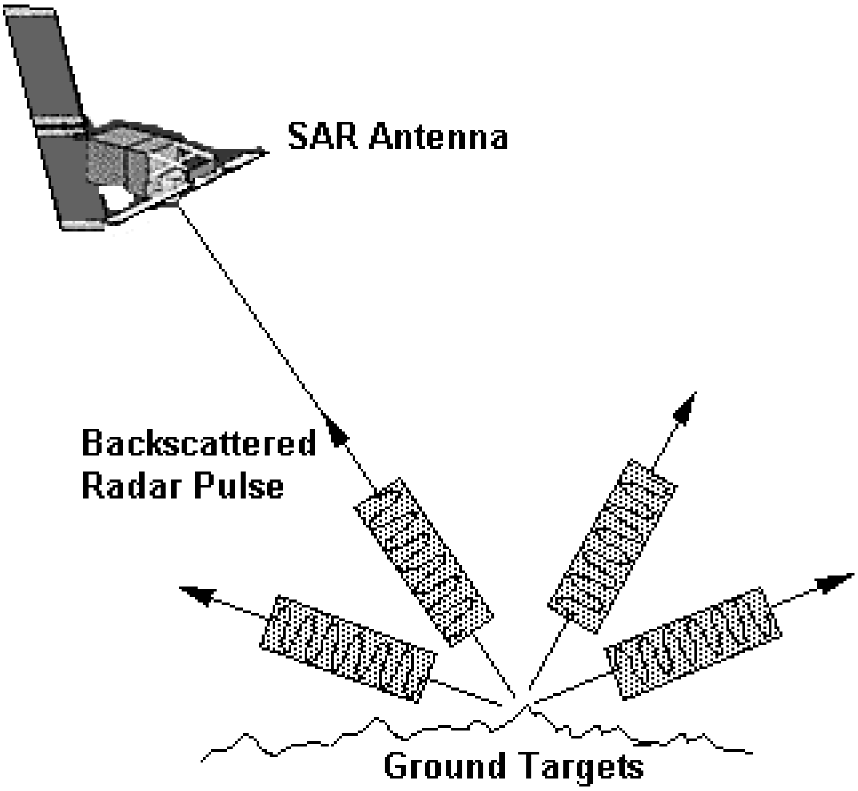


FIGURE 1.1: SAR satellite receiving backscattered pulses.

Creating one SAR image is done by sending radio wave pulses to the area of interest. The echo of each pulse is received by the satellite and recorded, as shown in Figure 1.1 (Liew, 2001b). All of the backscattered pulses gathered from one trip of the satellite back to its starting position are collected and used to form one image of the region of interest (*SAR Marine User's Manual*).

Satellite images can be used for many different applications. These applications usually fall under a larger category called remote sensing. Remote sensing refers to being alerted about changes in surfaces without having to be present at the surface location. SAR imagery can also be used for mapping out the surfaces of planet Earth and other planets. Some examples of remote sensing are measuring ecological changes along a coastline, or monitoring crop growth.

Radar signals have both an amplitude and a phase component. The images that we will discuss are SAR amplitude images. Amplitude refers to the energy of the reflected signal. This energy depends on the material of the surface being imaged. Hard objects, such as metal, backscatter more and thus the amplitude in these regions is higher than in others. This results in brighter spots in the image. Softer regions, such as grass, roads, etc., backscatter less of the incoming energy, so the amplitude is lower, and thus the image region appears darker. (*SAR Imagery* 2016)

In our project, we focus on remote sensing for the United States Department of Transportation (USDOT). Changes we would look for with satellite images would be changes in road quality, sinkholes, land slides, bridges, etc. Using imagery to observe changes in transportation infrastructure can save time as usually Department of Transportation (DOT) employees will have to traverse the roads in our network once a year to measure changes in road conditions. Often, large catastrophes cannot be prevented because we are not able to monitor areas remotely nor keep track of short-term changes over time. Remote sensing can help us monitor areas more consistently without sending out ground observers, as well as prevent disasters such as bridge collapse, etc. An example of a collapsed bridge is shown in Figure 1.2 (*Interstate 35W Bridge Collapse Critical Response*). We can see the destruction that this type of disaster can cause, with perhaps dozens of lives lost and extreme property and infrastructure damage that will cost large amounts of money to repair. Preventing these disasters before they occur using satellite imagery is our ultimate goal. Another more direct and measurable consequence of this work is the ability to more intelligently and effectively allocate DOT maintenance resources to where they are most needed.

Satellite imagery can be expensive, however, so when using images for remote sensing, we have to ensure accuracy of analysis of current conditions and accuracy of predictions. We cannot afford to make incorrect predictions and lose several millions of dollars by sending out ground personnel and equipment to fix a situation (such as a sinkhole) that has not actually occurred. Noise can greatly inhibit the ability of our analysis methods to accurately interpret the data, and, unfortunately, SAR images are very noisy. We have developed software tools that work with GIS (Geographic Information System) technology that raise an alarm if a bridge is moving (Figure 1.3), or if the road surface is rough and needs to be repaved. However, these toolboxes can sometimes raise false alarms because of the noisy nature of the data - we need to despeckle the data before we can use it to analyze ground conditions.

FIGURE 1.2: Catastrophic bridge collapse in Washington, U.S.

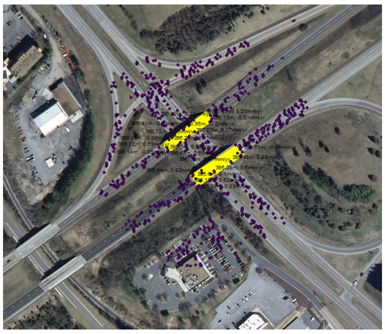


FIGURE 1.3: Warning alarm raised on bridges shown in ArcGIS.

The amplitude images that form our dataset are taken over Staunton, Virginia. The images are taken by Tele-Rilevamento Europa (TRE.) These images are taken over the period of one year, about once a week or so - the interval between images is not always the same. There are 67 images in our dataset. As these images are very large (over 600 MB each), we perform our smoothing on small sections of these images, which are 509 by 332 pixels, or 560 KB.

## 1.2 Noise

Noise is a random element that is often present in signals. It is an unwanted aspect of a signal that distorts our ability to analyze the data. There are many types of image noise, such as Gaussian noise, salt and pepper noise, and shot noise, to name a few. In images, we mainly differentiate between two classes of noise: additive and multiplicative. Multiplicative noise is often difficult to remove as it modulates the signal instead of just being added on top. Additive noise removal methods can be used to remove multiplicative noise by taking the log transform of the data - however, taking the log transform of an image is not lossless. It is more ideal to develop methods that can smooth raw data that is not log transformed.

SAR images are corrupted with a multiplicative noise type called speckle noise. Speckle noise is an inherent corruptive process of imaging using radar, SAR, and ultrasound techniques. The noise occurs because most surfaces that are being imaged are very rough smaller than the scale of the radio wavelength. The signal returning from the imaging surface to the transducer aperture can be modeled as an array of scatterers. These scatterers add constructively and destructively. The interference from these scatterers appears on the image as speckle noise, or dots on the image (Liew, 2001a). Because the noise is multiplicative, brighter regions in the image have noise values with higher intensity, and darker regions have noise values with lower intensity. To smooth these images, often an average of a certain number of "looks" is taken. One "look" is one period of the satellite's movement. If a few looks are averaged together, some of the noise can be decreased - however, there is still much to be desired in terms of noise removal. Also, any changes that occurred between looks will be lost during this averaging process.

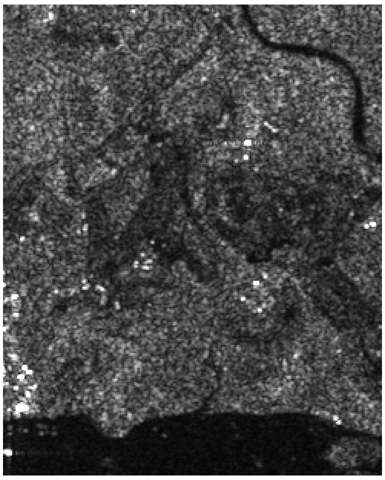


FIGURE 1.4: Example of speckle noise on a SAR image of a coastline.

### 1.2.1 Noise Removal

In general, noise distorts image content by changing the gradient of pixel intensities. This can be very degrading to an image because one of the main features we like to extract from images are the edges. The edges are extracted based on image gradients - or the change in pixel intensities as we move across the image. If the change in pixel intensity is very high, then we presume that there is an edge at that location. When there are noisy values in an image, spurious edges are detected which leads to distortion of the information in the image.

Most noise removal methods rely on smoothing. Smoothing algorithms make the sharp changes in pixel intensities more subtle. As a result, the noisy pixels begin to take on intensities closer to those in the surrounding regions of the image. However, blurring pixel gradients can lead to loss of edges. There is often a tradeoff between noise removal and edge preservation. We will show that our method removes noise yet retains important edges of roads and other objects.

One popular noise removal technique is called anisotropic diffusion. Anisotropy refers to not treating all regions of an image the same way. Isotropic diffusion, for example, will smooth an entire image at the same rate throughout, regardless of object details or edges. Anisotropic diffusion will slowly diffuse the image but will treat regions in the image with many edges differently from homogeneous regions. To clarify, regions with heavy edges will be passed with an all pass filter, while regions with few edges will be blurred with a low pass filter. The noise in regions with few edges will be removed using this low pass filter.

However, as discussed earlier, noise obscures edge content. We want to introduce a better way of detecting where there are edges in an image - this method will be discussed more fully in Chapter 3. Anisotropic diffusion is best used on images corrupted by additive noise, although there is an approach adapted to speckle noise that is of interest.

Speckle reducing anisotropic diffusion, or SRAD, is the speckle reducing algorithm that our method is derived from. Yu and Acton, 2002 This method works on the raw, not log transformed amplitude data. Our method, called Distance-Driven SRAD, or DD-SRAD, extends the SRAD equation to a temporal stack of images. Instead of smoothing each image in the stack separately using SRAD, DD-SRAD uses all of the images in the stack to estimate more accurately where edges are in the image. Our method provides improvement over the original SRAD approach in the context of despeckling a stack of images.

There are many other different methods of speckle noise removal which will be discussed more in depth in Chapter 2. However, there are few methods of noise removal which operate on a stack of images and retain the integrity of each image in the stack. Our method is one of the first methods to smooth an entire stack of SAR images - using the temporal information, but still preserving this information.

## 1.3 Problem

The problem we endeavor to solve is smoothing a time-varying stack of SAR images using temporal information without losing any of the individual images in the stack. Our method should reduce noise, retain edges, and preserve temporal information with better performance than the state of the art smoothing algorithms, as evidenced by lower mean squared error and higher peak signal-to-noise ratio.

## 1.4 Goals

The goals of this project are threefold. First, we want to show that our method removes more noise than other leading methods. Smoothing using our method leads to a lower mean squared error and higher peak signal to noise ratio than using other methods. Secondly, we hope to show the reader that our method retains information in the time-domain better than similar methods. Finally, we will show analytically that our method has lower overall complexity than the leading method for smoothing SAR image stacks.

## 1.5 Overview

In this chapter, we have discussed the background of SAR imagery and speckle noise, and touched on why smoothing of speckle noise in satellite imagery is important. Chapter 2 will be an overview of state of the art denoising methods. Our methodology for denoising will be covered in Chapter 3, and results from this and other methods will be displayed and assessed in Chapter 4. Chapter 4 will include discussion of Goals 1 and 2. Analysis of chosen methods will be shown in Chapter 5, with discussion of Goal 3. We will conclude our work in Chapter 6.

# Chapter 2

# Literature Review

## 2.1 Anisotropic Diffusion

Anisotropic diffusion is a popular noise removal technique used to smooth images corrupted by additive noise. The approach was first developed by Perona and Malik in 1987 (Perona and Malik, 1987). The continuous equation for this approach is outlined in Equation 2.1.

$$\begin{cases} \frac{\partial I}{\partial t} = div\left[c\left(|\nabla I|\right) \cdot \nabla I\right] \\ I\left(t=0\right) = I_0 \end{cases} \tag{2.1}$$

In Equation 2.1, $\nabla I$ is the gradient of the image, | | is the magnitude, *div* is the divergence, $I_0$ is the input image, and *c(x)* is a diffusion coefficient, defined in one of two ways:

$$c(x) = \frac{1}{1+(x/k)^2} \tag{2.2}$$

$$c(x) = exp[-(x/k)^2] \tag{2.3}$$

The magnitude of the gradient is used as an edge detector within *c(*$|\nabla I|$*)*. As $|\nabla I|$ » *k*, then *c(*$|\nabla I|$*)* $\rightarrow$ *0*, and filtering becomes allpass. Similarly, as $|\nabla I|$ « *k*, then *c(*$|\nabla I|$*)* $\rightarrow$ *1*, and filtering becomes Gaussian, or isotropic.

A discrete form of Equation 2.1 is shown in Equation 2.4.

$$I_s^{t+\Delta t} = I_s^t + \frac{\Delta t}{|\overline{\eta}_s|} \sum_{p\epsilon\overline{\eta}_s} c(\nabla I_{s,p}^t) \nabla I_{s,p}^t \tag{2.4}$$

$I_s^t$ is the image sampled discretely and *s* is the pixel position in *(x,y)*. $\Delta t$ is the time step, and $\overline{\eta_s}$ is the window around *s*. |$\overline{\eta_s}$| is simply the size of the window. $\nabla I_{s,p}^t = I_p^t - I_s^t, \forall\, p\, \epsilon\, \overline{\eta_s}$.

Anisotropic diffusion is an effective noise removal method, particularly for additive noise types. Diffusion is an iterative noise removal process - often the input image is smoothed slowly over time, for upwards of 100 iterations. For speckle noise, however, anisotropic diffusion can *enhance* the noise, because the gradient magnitude may treat the noise as an edge. We now move on to discuss filters specifically designed for speckle noise.

## 2.2 Adaptive Speckle Filters

There are a number of filters used in the SAR community specifically designed to handle speckle noise. Some of these filters include the Lee, Frost, and Kuan filters. The Kuan filter is similar to the Lee filter (D. T. Kuan and Chavel, 1987). All of these filters rely on a coefficient of variation, or a measure of how the intensity varies within a window (whether variation is caused by noise or signal.)

### 2.2.1 Lee Filter

The Lee filter is a windowing filter. In essence, the Lee filter computes a linear combination of the average intensity within a window and the value of the center pixel. The center pixel is then replaced with this linear combination. The filter is adaptive because averaging occurs in homogeneous regions, but the identity filter is applied in regions with edges. The coefficient of variation is low in homogeneous regions, and high in regions with sharp features (Lee, 1980).

The Lee filter assumes linear speckle noise and minimizes mean square error. The filter equation is

$$\hat{I}_s = \bar{I}_s + k_s(I_s - \bar{I}_s). \tag{2.5}$$

$\bar{I}_s$ is the average value within the window ($\eta_s$.) $k_s$ is the adaptive filter coefficient

$$k_s = 1 - C_u^2/C_s^2. \tag{2.6}$$

This is the adaptive filter coefficient. The two forms that $C_u^2$, which is constant for an input image, take are:

$$C_u^2 = 1/ENL \tag{2.7}$$

or

$$C_u^2 = \frac{var(z')}{(\bar{z}')^2}. \tag{2.8}$$

*ENL* is defined as the effective number of looks of the input image, and $z'$ is a homogeneous region of the image.

$C_s^2$ operates similarly to the diffusion coefficient in 2.2 or 2.3 and is defined as

$$C_s^2 = (1/|\eta_s|) \sum_{p\epsilon\eta} (I_p - \bar{I}_s)^2/(I_p - \bar{I}_s)^2. \tag{2.9}$$

If $C_s$ approaches $C_u$, then $k_s$ approaches zero, leading to a mean filter. If $C_s$ approaches $\infty$, then $k_s$ approaches 1, which is the edge case. This results in little to no change on or near edges.

There is an edge-enhancing modification of the Lee filter, which only includes pixels that seem to be from the same region in calculating local statistics (Ju and Moloney, 1997). However, this filter still suffers from the drawbacks of windowing, which will be discussed in Section 2.2.3.

### 2.2.2 Frost Filter

The Frost filter is similar to the Lee and Kuan filters in that there is a balance between all-pass filtering and averaging. The Frost filter incorporates a filter kernel that has the shape of an exponential distribution. This exponentially shaped filter can vary from an all-pass filter to a Gaussian filter on an adaptive basis.

The filter equation is

$$\hat{I}_s = \sum_{p \epsilon \eta_s} m_p I_p. \tag{2.10}$$

$m_p$ is

$$exp(-KC_s^2 d_{s,p}) / \sum_{p \epsilon \eta_s} exp(-KC_s^2 d_{s,p}) \tag{2.11}$$

and $d_{s,p}$ is

$$\sqrt{(i - i_p)^2 + (j - j_p)^2}. \tag{2.12}$$

*(i,j)* are the coordinates of the pixel *s* ($(i_p, j_p)$ for pixel *p*), and *K* is the damping factor.

Similar to the Lee filter, as $KC_s^2 \to 0$, averaging is achieved. As $KC_s^2 \to \infty$, there is no filtering (at edges.)

### 2.2.3 Drawbacks of Adaptive Speckle Filters

There are a number of drawbacks to using any of these speckle filters. First, these filters do not enhance edges. They only prevent smoothing near or over edges. Furthermore, since filtering becomes all-pass in the presence of an edge, any noise along an edge or on top of an edge will not be removed. This is particularly problematic when considering transportation applications of SAR imagery. Roads are the main edges in many of the SAR images, and if the roads remain noisy, then the amplitude data on the road cannot be assessed for road quality.

Secondly, these filters are non-directional. Near an edge, smoothing is completely inhibited. However, an approach that makes more sense is to only inhibit smoothing perpendicular to the edge, so that the edge is not smoothed over. Smoothing parallel to the edge, on image pixels running alongside the edge should not distort edge content.

Thirdly, there are hard thresholds used in all of the above filters. These thresholds may lead to artifacts produced by averaging (blotchy patches) and also noisy edges because of strict all-pass filtering.

Lastly, these filters are all window-based approaches, and thus are very dependent on window size and type. Larger windows can lead to oversmoothing, which registers as a blurred effect on the image. Smaller windows may not smooth enough and can leave a large amount of noise behind. Choosing the correct window size for each application may be a difficult problem to optimize. As for window shape, most common approaches will use a square window. A square window leads to rounding of features that are found in different orientations than the square window.

Because of these many drawbacks, SRAD was developed, which is discussed in the next section, Section 2.3.

## 2.3 Speckle Reducing Anisotropic Diffusion

Speckle Reducing Anisotropic Diffusion, or SRAD, proposes a new partial differential equation (PDE) based on original anisotropic diffusion and the adaptive speckle filters. The SRAD PDE is as follows:

$$\begin{cases} \partial I(x,y;t)/\partial t = div[c(q)\nabla I(x,y;t)] \\ I(x,y;0) = I_0(x,y), (\partial I(x,y;t)/\partial \overrightarrow{n})|_{\partial\Omega} \end{cases} \tag{2.13}$$

where $\Omega$ is a 2D coordinate grid where the image is non-zero. $\partial\Omega$ is the border of $\Omega$, and $\overrightarrow{n}$ is the outer normal to said border.

The divergence of $c(\cdot)\nabla I$ is calculated as:

$$d^n_{i,j} = (\frac{1}{h^2})[c^n_{i+1,j}(I^n_{i+1,j} - I^n_{i,j}) + c^n_{i,j}(I^n_{i-1,j} - I^n_{i,j}) + c^n_{i,j+1}(I^n_{i,j+1} - I^n_{i,j}) + c^n_{i,j}(I^n_{i,j-1} - I^n_{i,j})] \tag{2.14}$$

which leads us to the SRAD update function:

$$I^{n+1}_{i,j} = I^n_{i,j} + \frac{\Delta t}{4} d^n_{i,j} \tag{2.15}$$

As in the original anisotropic PDE, the diffusion coefficient can take one of two forms:

$$c(q) = \frac{1}{1 + [q^2(x,y;t) - q_0^2(t)]/[q_0^2(t)(1 + q_0^2(t))]} \tag{2.16}$$

$$c(q) = exp\left\{-[q^2(x,y;t) - q_0^2(t)]/[q_0^2(t)(1 + q_0^2(t))]\right\} \tag{2.17}$$

The diffusion coefficient for the SRAD equation is composed of two parts: the instantaneous coefficient of variation (*q(x,y;t)*), and the speckle scale function ($q_0(t)$). We can think of the instantaneous coefficient of variation, or ICOV, as the coefficient of variation as we shrink our window of interest to a single point.

The ICOV is defined as:

$$q(x,y;t) = \sqrt{\frac{(1/2)(|\nabla I|/I)^2 - (1/16)(\nabla^2 I/I)^2}{[1 + (1/4)(\nabla^2 I/I)]^2}} \tag{2.18}$$

and it contains both a gradient magnitude term and a Laplacian term. The ICOV allows for edge detection in bright and dark regions because of the addition of the Laplacian term. The speckle scale function is defined as:

$$q_0(t) = \frac{\sqrt{var[z(t)]}}{\overline{z(t)}} \tag{2.19}$$

where *z(t)* is a homogeneous region. This homogeneous region can be defined by the user manually. Often, because manual selection takes time, the user may choose to use the default setting for a homogeneous region. This default is a hard-coded set of coordinates, and so mimicks a random selection of a region from an image. This default homogeneous region may not even give a homogeneous region for every image that we use SRAD on, let alone the most homogeneous or the largest homogeneous region.

SRAD is an improvement on both the original anisotropic diffusion algorithm and on the adaptive speckle filters. At the midpoint of an edge, the Laplacian term in Equation 2.18 undergoes a zero-crossing. This insures that diffusion only occurs along the contour of the edge, which is an improvement on adaptive filtering, where there is no smoothing along the edge contour. However, there is also a "negative" diffusion that occurs perpendicular to the edge. This is not present in simple anisotropic diffusion and is a phenomenon of SRAD. This "negative" diffusion leaves the dark regions of edges darker and the brighter regions brighter - essentially, SRAD enhances contrast along edges. The edges in the resulting smoothed images are more prevalent (Yu and Acton, 2002).

## 2.4 Space Adaptive Filtering: DespecKS Algorithm

This despeckle algorithm was developed in 2011 for use on SAR amplitude image stacks (A. Ferretti and Rucci, 2011). Space adaptive filtering refers to changing the filter parameters depending on position in the image. The benefit of using a space adaptive filter is that these filters can remove noise while still preserving image detail. Some of the methods we have described in the previous sections can also be considered space adaptive filters.

To smooth the image stack using a space adaptive filter, there must be a way to decide whether two pixels are statistically homogeneous pixels (SHP). We can think of our image stack as concatenated data vectors of intensity values. $d(P)$ is defined as:

$$d(P) = [d_1(P), d_2(P), ..., d_N(P)]^T \tag{2.20}$$

and is a vector of intensity values for arbitrary pixel $P$. $d_1(P)$ is the pixel value in the first image, etc. $N$ is the number of images in the stack. To test whether two data vectors ( $d(P_1)$ and $d(P_2)$, for example) are from the same continuous distribution, we can use the Kolmogorov-Smirnov (KS) test. This test will test the null hypothesis that pixels $P_1$ and $P_2$ are from the same distribution. If this null hypothesis cannot be rejected at a particular $\alpha$ significance level, then the pixels $P_1$ and $P_2$ will be considered statistically homogeneous.

To perform the KS test on pixel vectors, a cumulative distribution function (CDF) must be built from the probability distribution function (PDF) of each pixel's sorted intensity values. The test determines whether the amplitude data for the pixels are drawn from the same PDF. If we define **x** = | **d** |, then we can write an unbiased estimator for the CDF of a pixel, $S_N(X)$, as

$$\begin{cases} 0, & \text{if} \quad X < x_1 \\ \frac{k}{N}, & \text{if} \quad x_k \leq X < x_{k+1} \\ 1, & \text{if} \quad X \geq x_N. \end{cases} \tag{2.21}$$

$x_i$ is the $i^{th}$ element in the list of pixel intensities.

The two sample KS test measures the maximum absolute difference between the cumulative distribution functions $S_N^{p1}$ and $S_N^{p2}$. This difference, which can also be used as a distance metric, is defined by

$$D_N = \sqrt{N/2} \quad \text{sup}_{x \epsilon R} |S_N^{p1}(x) - S_N^{p2}(x)|. \tag{2.22}$$

The distribution of $D_N$ is approximated by

$$P(D_N \leq t) = 1 - 2\sum_{n=1}^{\infty}(-1)^{n-1}e^{-2n^2t^2} \tag{2.23}$$

which is the CDF of the Kolmogorov-Smirnov distribution, *H(t)*. It should be noted that this CDF does not depend on the CDF of the input data. The two vectors can be said to belong to the same distribution if $D_N \leq c$. The threshold *c* is dependent on the chosen significance level $\alpha$, where

$$\alpha = 1 - H(t). \tag{2.24}$$

The DespecKS algorithm (A. Ferretti and Rucci, 2011) involves the following steps:

1. For image 1 in the stack, move a sliding window (15 x 21) across the image. *P* is the center pixel of the window.

2. All pixels in the estimation window are compared to *P* using the two-sample KS test, given $\alpha$. Pixels that pass the KS test (fail to reject the null hypothesis) are labeled SHP.

3. Discard SHP that are not connected to *P*, directly or through other SHP.

4. Intensity values for SHP are averaged spatially, and the value of *P* is replaced with the SHP average, for each image in the stack.

A drawback to this method are that there is an unweighted averaging of all SHP pixels. One could argue that although we cannot reject the hypothesis that $P_1$ is statistically homogeneous with *P*, it may be more likely that $P_2$ is drawn from the same distribution as *P* than the probability that $P_1$ is. For this reason, it may be dangerous to average all SHP together, disregarding the distance $D_N$ from *P*. For this reason, we use the KS distance, and other distances in our smoothing algorithm, instead of the KS test combined with averaging.

Another drawback is that the KS test has poor sensitivity in the tails of the distribution to cases where the null hypothesis is untrue. Using the KS distance will solve this problem, and we also test other distances to determine which ones perform better.

There is no one best way to solve the speckle denoising problem, as evidenced by the many methods described above. We proceed to experiment in the diffusion realm. We introduce a number of new methods which improve denoising performance.

# Chapter 3

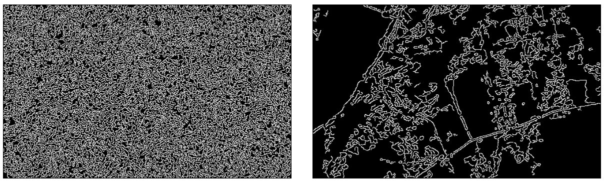
# Methods

## 3.1 Median Driven Diffusion

Anisotropic diffusion and its other derivative forms use the edges of an image as a basis for smoothing. Where the gradient magnitude is high, we assume there is an edge and the diffusion equation behaves as an all-pass filter. If the gradient magnitude is low, then we perform Gaussian filtering. But this approach depends on the ability of the gradient magnitude to determine edges. This approach can often relay many false positives, especially when the noise content is high. A high gradient magnitude may signify a noisy region. If we all-pass filter this region, then the noise will not be removed from the image.

In contrast, we propose a new idea using a time-series. Earlier we have mentioned that we often use averaging to produce one smooth image from a stack of noisy images. However, using this approach leaves us with one image where all of the temporal information has been averaged together and is no longer discernable. Instead, we would like to use this smoothed image to drive our diffusion process.

Instead of calculating the gradient magnitude on the original noisy image, we take the median of the temporal stack and calculate the gradient magnitude on the median image. In Figure 3.1, we see edgemaps of both one single SAR image and the median of a stack. The single SAR image has many edges due to noise - there are no real discernable features. The median edgemap has delineation of where there are roads, and perhaps other image features, such as buildings. The median edgemap still has edges due to noise, but the overall number of spurious edges detected are visibly fewer. The amount of noise removed depends on the size of the stack. Our SAR image stacks are 67 images long, so the temporal median removes more noise than on a stack with only 20 images. Since the median image has better defined edges than the noisy image, the gradient magnitude of the median image should be more representative of real edges than of noise. We use this assumption to build our two methods based on the median.



(A) Edgemap of SAR (B) Edgemap of median

FIGURE 3.1: Comparing edges of Original SAR vs. Median

*Median Driven Anisotropic Diffusion:* For this approach, we calculate the gradient magnitude on the median of the image and use this coefficient of diffusion in the anisotropic diffusion equation (2.1).

*Median Driven SRAD:* We use the same coefficient of diffusion as in Median Driven Anisotropic Diffusion in the SRAD equation (2.13).

## 3.2 Distance Driven SRAD

Following the development of Median Driven SRAD, we decided to try and improve upon our method. Inspired by the DespecKS method from Section 2.5, we use the idea of a KS distance between pixel time series to drive our diffusion. Instead of using the edges of the median to diffuse the noisy images, we propose a new coefficient of diffusion:

$$q(x,y;t) = \sqrt{\frac{(1/2)(\sum D^2/I^2) - (1/16)(\sum D/I)^2}{[1+(1/4)(\sum D/I)]^2}}. \tag{3.1}$$

This diffusion coefficient depends on the distance between neighboring pixel time series, $D$. $D$ replaces the gradient magnitude in the original SRAD coefficient of diffusion. For instance, if there are two pixels on one section of a road, we can assume that the intensity changes of the two pixels should be similar through time. The distance, or difference between the time series of these pixels should be minimal. However, the difference between the behavior of a pixel in a field next to the road and the pixel on the road may be very divergent over time. The road may be paved at different times of the year, leading to changes in intensity of the road pixel. A field pixel may change intensity based on vegetation growth, cutting of grass, etc. As the time series of these pixels behave differently through time, the distance between them will be larger than that of the two road pixels. This large distance can be seen visually as an edge: the edge delineated by the boundary of the road against the field. Just as gradient magnitude can denote edges, thus distance between neighboring pixel series can also mark edges. We use this edge detector as the basis for driving our Distance Driven SRAD, or DD-SRAD. The idea was inspired by a similar use of distance in (A. Ferretti and Rucci, 2011).

There are a number of distances that one can use with DD-SRAD - we have implemented and tested three different distance metrics.

### 3.2.1 Root Sum of Squares

This is a simple distance metric, similar to root mean square error (RMSE.) The two pixel time series are differenced, with one difference value for each time instance in the stack. This difference vector is squared to retain positive values. The squared differences are summed, and the square root is then taken of the sum. This provides a Root Sum of Squares (RSS) difference metric for two time series, which can be compared to other distance metrics.

### 3.2.2 Kolmogorov-Smirnov

The Kolmogorov-Smirnov (KS) distance is a distance measured between the empirical distribution functions of two time-series. An empirical distribution function is built for each pixel in the temporal stack. Neighboring pixels are differenced, using the largest two-point distance between the points in the distribution functions. This distance can be between any two points in the distribution function. The distance that we use is often referred to as the Kolmogorov-Smirnov test statistic (Massey, 1951).

### 3.2.3 Bhattacharyya

The Bhattacharyya distance is defined by the following equation:

$$D_B(p,q) = -ln(BC(p,q)) \tag{3.2}$$

The Bhattacharyya coefficient is defined as the following:

$$BC(p,q) = \sum_{x \epsilon X} \sqrt{p(x)q(x)}. \tag{3.3}$$

This distance is particularly good for noisy data, as it can be thought of as the integral distance between two probability distributions, instead of a distance between just two points, as in the KS distance (Bhattacharyya, 1943).

### 3.2.4 Gaussian Weighted vs. Unweighted Distances

We first tried treating all distances as equal, regardless of relative position in the temporal stack. As a result, we sometimes noted ghost effects in the images: a remnant of an object in one denoised image when the object had not appeared in the original image. This was as a result of the object being present in another of the images, at a far away time in the temporal stack.

To correct for this ghosting, we introduce a weighting of distances. This consists of a Gaussian the length of the stack, shifted along the stack. For instance, if we are smoothing the first image in the stack, the Gaussian is centered on the first index in the stack. The images temporally closer to the first image (such as the second and third images) will have distances that are weighted higher than the distance between the first image and the tenth image. This way, changes in edges further away in time are not registered as edges in the current time instance. We combine Gaussian weighting with RSS distance. Results will be shown in Chapter 4.

## 3.3 Homogeneous Region Detection

SRAD, and the derivative forms of SRAD we discuss here, require a homogeneous region to be selected at initialization in order to calculate the statistics of the speckle noise. This homogeneous region can be selected at random, or by hand by the user. A random selection is not ideal, as the region selected may be non-uniform. A user selected region also presents many flaws: it is prone to error, may not be the largest possible homogeneous region, it is constrained to a rectangular window, and it is time consuming and arduous especially if one has to select a homogeneous region for each of a large stack (60 plus) of images.

Homogeneous regions in an image are characterized by regions where there are few or no edges. To detect the homogeneous region in the noisy images, we perform edge detection on the median image. Performing edge detection on a smooth image provides a more clear edge map, with less spurious edges detected due to noise. Then we perform a filtering, where if any pixel in a window has an intensity value correlating to an edge, then we color the whole window as an edge. This filtering is done to make the homogeneous region detection less sensitive to small edges found by the edge detection, so that there are large regions of homogeneity, instead of small regions broken up by edges due to noise. The optimal window size may vary for image stacks, so the user can view the homogeneous regions produced by varying window sizes, and choose the region that they are happy with. The window size ranges generally from 3x3 to 9x9. The larger window sizes are less prone to error - however, as the window size increases, the area of the homogeneous region detected decreases. After filtering, connected component analysis is performed, and the largest connected component (largest edge-free region) is taken as the homogeneous region. A minimum for this region is set to at least 400 square pixels. If the homogeneous region found is smaller than 400 square pixels, a random region is selected. However, in all of the experiments and datasets we have tried, the homogeneous region found has always been larger than 400 square pixels.

In Chapter 4, we will show results from using Median driven SRAD and from using DD-SRAD, in all its forms. Our methods will be compared to the some of the leading methods discussed in Chapter 3. We will also show the statistical significance of our homogeneous region detection algorithm.

# Chapter 4

# Results/Findings

In this chapter, we will discuss the results from our homogeneous region detection as well as from our smoothing algorithms. We will start by showing that our homogeneous region detector locates regions of lower variance in the noisy images than when using a random region selection. We will then test our smoothing algorithms against the state-of-the-art smoothing methods. We will compare the mean-squared error and peak signal-to-noise ratio after smoothing synthetic data sets, and will show that our methods have lower MSE and PSNR than the leading algorithms. We will also show that our methods have low variance in homogeneous regions post denoising. We will then show temporal change preservation by comparing MSE of the temporally changing feature. Our method has lower local MSE than a leading temporal smoothing method. We will finally compare our preservation of temporal information against the average of an image stack.

## 4.1 Homogeneous Region Detection

We wish to demonstrate the efficacy of our automatic selection of a homogeneous region. A homogeneous region should have low variability compared to a non-homogeneous region. We will compare the variance of our homogeneous region to a randomly selected region. We choose to compare to a randomly selected region because the original SRAD implementation uses an ad-hoc set of coordinates for choosing a homogeneous region - these region coordinates/dimensions do not change based on the image being smoothed, so the selection is, in essence, random. A homogeneous region should have a lower variance than a non-homogeneous region. In Figure 4.1, we have displayed a detected homogeneous region on the edgemap of the median of the sub-stack. There are few edge pixels within the boundary of the homogeneous region detected.

To test our method for detecting homogeneous regions, we generated 10 stacks of sub-images from the large SAR data stacks. Each of the sub-image stacks have 67 slices, and are approximately 300 pixels by 500 pixels. The sub-images were clipped from all over the original data, so the underlying terrain in each sub-image stack varies significantly from stack to stack.

To detect a homogeneous region for each sub-image stack, we ran our homogeneous region detection method on the stack to pick one homogeneous region. A window size of 3x3 was used for filtering for all sub-image stacks. This same homogeneous region was used for all slices in the sub-stack. The variance of this homogeneous region was calculated for each slice in the sub-stack.

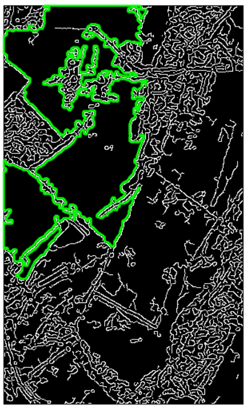


FIGURE 4.1: Homogeneous region displayed on edgemap of median.

The randomly selected region is chosen with approximately the same size as the detected homogeneous region for each sub-image stack. The randomly selected region is a square region, selected at random from within one image of the sub-stack. This same random region is used to clip all of the slices of the sub-stack. The variances of all of the random regions of all of the 10 sub-stacks are then calculated, 67 variances per sub-stack (corresponding to 67 slices), for both detected and randomly selected regions.

### 4.1.1 Statistical Testing

Our goal is to show that the variances of regions detected by our homogeneous region detection method are lower than those of the regions that are randomly selected. In Table 4.1, we can see some general statistics of the two sample sets. As we can see, the statistics for the regions detected by our method are always lower than the randomly selected region statistics. It is interesting to note that the mean and mode values for the variances are not so far apart as one might expect. However, the maximum variance values are extremely different, over 10 times greater for the randomly selected region. Also, the standard deviation for Random regions is much higher than that of Homogeneous Regions - showing that the local variance of Random regions has wide spread. It is important to note that while some randomly selected regions may have similar variances to detected homogeneous regions, there is a risk that the randomly selected region will have high local variance, and will thus be classified as inhomogeneous. If we use an inhomogeneous region to calculate the speckle scale statistic, the SRAD or DD-SRAD smoothing procedure will be corrupted.

First, we tested both distributions for normality, to see if a t-test or other tests for normal distributions would be suited. I used the Jarque-Bera test for both the variances of detected homogeneous regions and the variances of randomly selected regions. The Jarque-Bera test tests against the null hypothesis that the data in question is normally distributed with unknown mean and variance (Jarque and Bera, 1987). The null hypothesis can be

TABLE 4.1: Statistics for Variances of Randomly Selected vs. Homogeneous Regions

| **Statistic** | **Random** | **Homogeneous** |
|---|---|---|
| Min | 0.1015 | 0.0545 |
| Max | 5.1184 | 0.2603 |
| Mean | 0.4982 | 0.1230 |
| Std. Dev. | **0**.7458 | **0**.0333 |
| Median | .2113 | .1192 |
| Mode | .1015 | .0545 |

rejected at the 1% significance level, meaning that the variances for the regions are not normally distributed. Thus, we cannot use a simple t-test to compare the distributions of variances.

Since we cannot assume the variances are normally distributed, we use a two-sample Kolmogorov-Smirnov test to compare the two sets of variances. The two-sample Kolmogorov-Smirnov test tests against the null hypothesis that the two datasets are from the same continuous distribution. We test the two distributions of variances at the 1% significance level and reject the null hypothesis. The two variance sample sets are not from the same distribution.

Moreover, we can also test the alternative hypothesis that the empirical continuous distribution function (ECDF) of the randomly selected region variances is smaller than that of the detected homogeneous region variances. In general, if the ECDF of one sample set is smaller than another ECDF, then the data values of the sample set with the smaller ECDF are larger. The size of the ECDF has an inverse relationship with the magnitude of the data values in the sample set. The p-value for this test is 1.1666 x $e^{-144}$, which is a very small number. The p-value of a test is the probability that we observe the test statistic (calculated from the samples) under the null hypothesis. A small p-value indicates the low probability that such a test statistic could occur under the assumptions of the null hypothesis. The p-value for the Kolmogorov-Smirnov test is highly accurate for sample sizes that are large, and is reasonably accurate for:

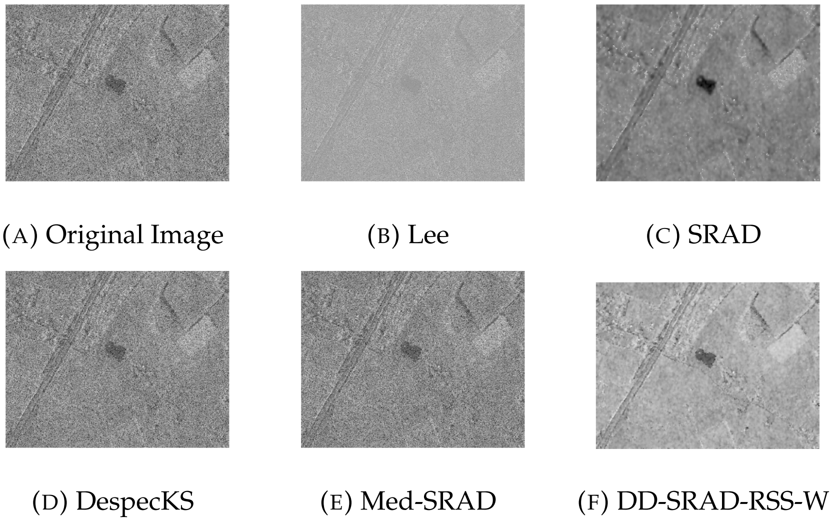
$$(n1 * n2)/(n1 + n2) \geq 4 \tag{4.1}$$

where *n1* and *n2* are the two sample sizes. *n1* and *n2* are both equal to 670 for our experiment, and so this equation holds (with equality at 335).

We reject the null hypothesis that the ECDF's are equal in favor of the alternative hypothesis at the 1% significance level. Therefore, the ECDF of the random region variances is smaller than the ECDF of the detected region variances, and the variance values of the random regions are *larger* than the values of the detected homogeneous regions.

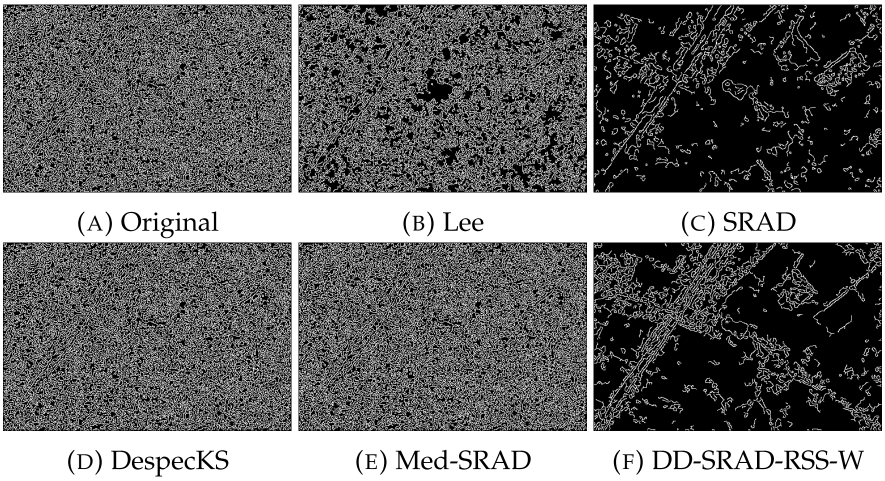
## 4.2 Smoothing Algorithms

In this section, we will test our smoothing algorithms and other leading smoothing algorithms to compare results against our goals (see Chapter 1.)


(A) Original Image (B) Lee (C) SRAD

(D) DespecKS (E) Med-SRAD (F) DD-SRAD-RSS-W

FIGURE 4.2: Comparing Images from Smoothed SAR Stacks


(A) Original (B) Lee (C) SRAD

(D) DespecKS (E) Med-SRAD (F) DD-SRAD-RSS-W

FIGURE 4.3: Smoothed SAR Stack Edgemaps

We show some preliminary smoothing results on a SAR stack of 67 images in Figure 4.2. In Figure 4.2f, we see a result using DD-SRAD - this image is the smoothest while also maintaining clear edges. We also show the resulting edgemaps in Figure 4.3. There are more edges in Figure 4.3f than in Figure 4.3c, but this may be beneficial as a secondary road is preserved in Figure 4.2f that is disconnected in Figure 4.2c. However, we must find a way to quantify the performance of DD-SRAD compared to other methods, so we run experiments on synthetic datasets.

### 4.2.1 Synthetic Datasets

To analyze our methods based on the goals, we test the methods on synthetic datasets. We do this because many of the metrics used to evaluate our methods, such as mean squared error (MSE), rely on comparing the smoothed image to the original ground truth image (with no noise.) As we do not have ground truth images for our satellite data, we simulate data with similar features and add noise to this synthetic data. Then we can compare the smooth results from our methods to the original, non-noisy images.

Noise is added to our synthetic images by generating speckle noise from a Rayleigh distribution. A Rayleigh distribution can be formed from two independent Gaussian random variables, with mean 0 and variance $\sigma^2$. If we call these random variables $X \sim N(0, \sigma^2)$ and $Y \sim N(0, \sigma^2)$, then the random variable $R$ is Rayleigh distributed ($R$ ∼Rayleigh($\sigma$)) where $R = \sqrt{X^2 + Y^2}$ (Hogema, 2005).

We generate two Gaussian random variables and combine them to form a Rayleigh random variable. We then multiply our base synthetic images with this Rayleigh variable to create multiplicative speckle noise. We use a noise variance randomly chosen between .08 and .09 when generating the Gaussian random variables - after using Immerkaer's fast noise estimation method on our SAR images, we find that the estimated noise variance is between .08 and .09 for each image, so we use this noise variance for our synthetic images as well (Immerkaer, 1996).

We generate two synthetic, noisy datasets. The dataset sizes are variable, from 11 to 17 images. To simulate variation over time, we introduce features at different points during the time-series, and change the intensity of these objects over time. In our synthetic data, straight lines represent roads, and circular or ovular objects represent bodies of water. Rectangular objects are buildings.

### 4.2.2 Methods Tested and Parameters Used

#### Lee Filter

For the Lee Filter, we used a window size of 3. This window size refers to the size of the window used for averaging. The other parameter is the coefficient of variation in a homogeneous region. As the original Lee filter does not include automatic homogeneous region detection, we select a random region (40 pixels by 40 pixels) as a homogeneous region and calculate statistics of this area.

#### SRAD

We ran the SRAD algorithm on all three datasets for 200 iterations, with a smoothing time-step ($\triangle t$) of .05. A small smoothing time-step smoothes a minimal amount with each iteration, which is why there are so many iterations performed. Using too large of a time-step in combination with fewer iterations is not preferred, as this can negatively affect the stability of diffusion algorithms. The homogeneous region chosen is the same randomly selected 40x40 region used in with the Lee Filter.

#### DespecKS

We test a version of the DespecKS algorithm, written by ourselves. For the DespecKS algorithm, we use a 15x21 window to test for SHP. The alpha significance value is set to .05.

### Median Driven SRAD

Median Driven SRAD is also driven for 200 iterations, with step-size .05. The same homogeneous region is used as in SRAD and Lee filtering.

### DD-SRAD

All DD-SRAD algorithms are run for 200 iterations. The time-step used for DD-SRAD is also .05. For all DD-SRAD algorithms, we use our homogeneous region detection algorithm to specify a homogeneous region, using window-size 3x3.

*DD-SRAD-KS:* DD-SRAD-KS stands for Distance-Driven SRAD run using KS distance.

*DD-SRAD-B:* DD-SRAD-B stands for Distance-Driven SRAD using Bhattacharyya distance.

*DD-SRAD-RSS:* DD-SRAD-RSS uses RSS distance, weighted evenly across all images regardless of position in the stack.

*DD-SRAD-RSS-W:* DD-SRAD-RSS-W refers to DD-SRAD using RSS distance, but using the weighted approach mentioned in Chapter 3.

## 4.2.3 Goals

The two goals mentioned in Chapter 1 are reintroduced in this section.

### Goal 1 - Noise Removal

The first goal is to show that DD-SRAD has a lower average MSE and PSNR than other leading methods. MSE is mean squared error - it is calculated by subtracting the ground truth image from the smooth image, to determine how many errors there are in the smooth image. This difference is squared and averaged across the images to get an MSE for each stack. The average MSE reported in Table 4.3 is averaged across the two synthetic datasets.

PSNR, or peak signal-to-noise ratio is implemented using MSE:

$$PSNR = 10log_{10}(peakval^2/MSE). \tag{4.2}$$

*peakval* is the largest value allowed by the data; the synthetic datasets range from 0 to 1, so *peakval* here is 1. PSNR is measured in decibels. A higher value of PSNR is more desirable. In Table 4.3 we compare the MSE and average PSNR for all methods tested.

Med-SRAD shows improvement upon the SRAD algorithm. Med-SRAD has lower MSE than SRAD, and higher PSNR. Med-SRAD performs the best out of all algorithms for Experiments 1 and 2. We will observe further metrics to compare performance to the DD-SRAD methods.

Within DD-SRAD, DD-SRAD-B has the highest MSE and lowest PSNR. DD-SRAD-KS, DD-SRAD-RSS, and DD-SRAD-RSS-W all perform with similar MSE and PSNR values. DD-SRAD has lower MSE and higher PSNR than both the Lee filter and the SRAD approach, which are representative

TABLE 4.2: Average MSE and PSNR: Experiments 1 and 2

| **Method** | **MSE** | **PSNR** |
|---|---|---|
| Lee | .5902 | 2.2927 |
| SRAD | .5930 | 2.2746 |
| DespecKS | .6004 | 2.218 |
| **Med-SRAD** | .3607 | 4.4373 |
| **DD-SRAD-KS** | .5216 | 2.8319 |
| **DD-SRAD-B** | .5629 | 2.5020 |
| **DD-SRAD-RSS** | .5240 | 2.8115 |
| **DD-SRAD-RSS-W** | .5218 | 2.8303 |

TABLE 4.3: Mean and Standard Deviation for Homogeneous Regions: Experiments 1 and 2

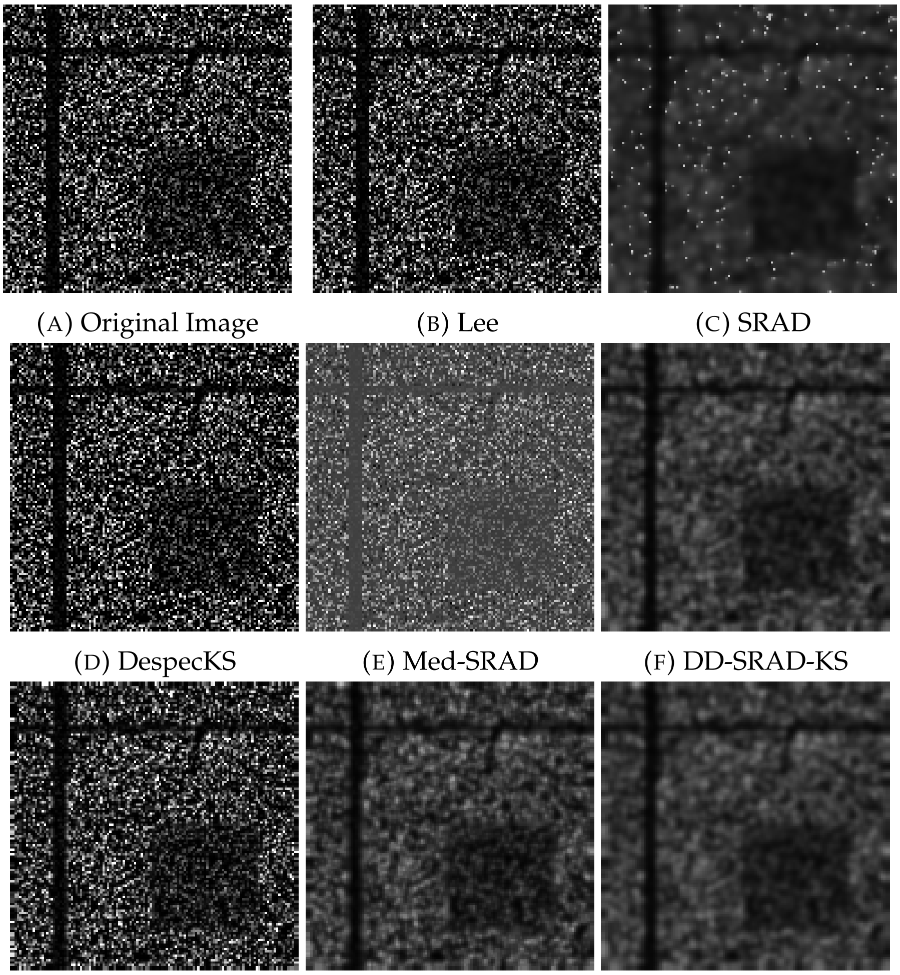
| | **Experiment 1** | | **Experiment 2** | |
|---|---|---|---|---|
| **Method** | **Mean** | **Std. Dev.** | **Mean** | **Std. Dev.** |
| Lee | .1019 | .1427 | .0584 | .0958 |
| SRAD | .0814 | .0286 | .0474 | .0276 |
| DespecKS | .1019 | .1553 | .0584 | .1030 |
| **Med-SRAD** | .3344 | .0741 | .2659 | .0535 |
| **DD-SRAD-KS** | .1097 | .0378 | .0827 | .0343 |
| **DD-SRAD-B** | .1078 | .1396 | .0811 | .0784 |
| **DD-SRAD-RSS** | .1088 | .0546 | .0823 | .0402 |
| **DD-SRAD-RSS-W** | .1102 | .0348 | .0833 | .0332 |

of speckle reduction techniques used by the SAR community. DD-SRAD also has lower MSE and higher PSNR than the DespecKS method, which is representative of speckle removal on temporal stacks.

We have measured the standard deviation and mean values of homogeneous regions for Experiments 1 and 2, shown in Table. These are averaged over the stack. We prefer a lower standard deviation for homogeneous regions. In the original, speckle-free image, a homogeneous region should have little variation, so most variation is due to noise. After smoothing, if a region has little variation, then we assume that the noise has been removed (Yu and Acton, 2002).

For both experiments, the standard deviations of homogeneous regions is lower for DD-SRAD-KS, DD-SRAD-RSS, and DD-SRAD-RSS-W than for Med-SRAD. DD-SRAD-B does not have as low of a standard deviation as its counterparts. We see that SRAD has the lowest standard deviation for homogeneous regions, compared to the other methods. However, when we observe images after smoothing, the flaws of SRAD become clear. In Figure 4.4, we see in 4.4c that the SRAD algorithm has left many high intensity speckle artifacts behind. However, Figures 4.4f, 4.4h, and 4.4i have smoothed these bright noisy pixels.

The distance metrics analyze pixel behavior through time, unlike the



(A) Original Image (B) Lee (C) SRAD

(D) DespecKS (E) Med-SRAD (F) DD-SRAD-KS

(G) DD-SRAD-B (H) DD-SRAD-RSS (I) DD-SRAD-RSS-W

FIGURE 4.4: Comparing results from Experiment 2

SRAD ICOV, which only uses spatial analysis. Using temporal analysis allows us to compare the bright pixels to neighboring pixels - if the bright pixels follow the same behavior as neighboring pixels, then the bright speckle noise is smoothed. The SRAD algorithm cannot perform this type of temporal analysis, so bright speckle remains in the SRAD despeckled images.

### Goal 2 - Time Variation Preservation

To measure whether our algorithms preserve time varying information, we compare our algorithms with the DespecKS algorithm, the only other algorithm that also takes advantage of the image stack. For each synthetic dataset there is only one feature that appears or disappears over time. We form a rectangular bounding box around this feature and calculate the MSE for this region between smoothed images and the ground truth image. These MSE's are reported in Table 4.4.

We find that all DD-SRAD methods have lower MSE to the DespecKS algorithm, showing that DD-SRAD preserves temporal information better than the DespecKS algorithm. Using a weighted RSS distance produces lower error than using a non-weighted RSS distance. However, at times, other distances, such as the KS distance in Experiment 2, outperforms the

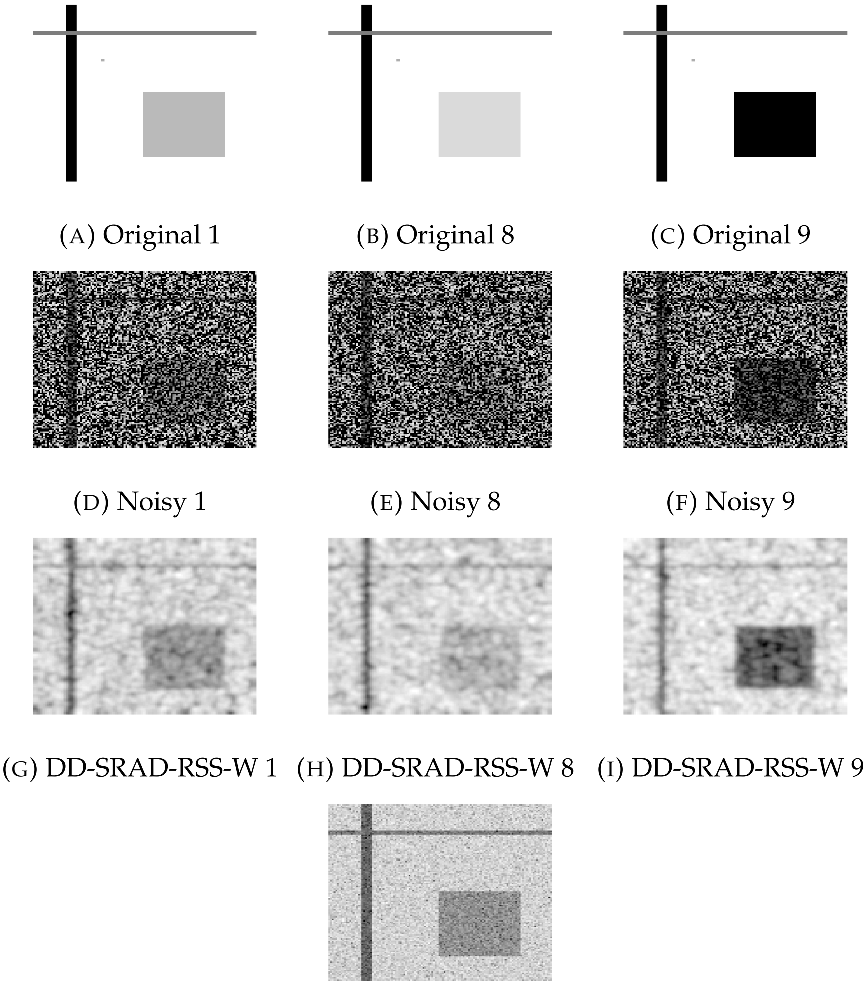
TABLE 4.4: MSE of Time-Varying Structure

| **Method** | **Experiment 1** | **Experiment 2** |
|---|---|---|
| DespecKS | .6158 | .4263 |
| **DD-SRAD-KS** | .5548 | .3685 |
| **DD-SRAD-B** | .6030 | .4019 |
| **DD-SRAD-RSS** | .5571 | .3703 |
| **DD-SRAD-RSS-W** | .5551 | .3686 |

weighted RSS distance. If runtime of algorithms is a concern, then using weighted RSS distance is faster than using either KS distance or Bhattacharyya distance, and results are comparable, or better, in terms of MSE.

The difference between performance in distances may be a result of using unscaled distances. The Bhattacharyya distance values are much larger than the distances calculated using the two-sample KS test, or when using RSS distance. As a result, smoothing is inhibited to a greater extent when using Bhattacharyya distance. In future experiments, distances metrics need to be scaled for comparison.

We also show change preservation on one of our time-varying datasets in Figure 4.5. We show images 1, 8, and 9 from the stack. Our algorithm, DD-SRAD-RSS-W is able to preserve the intensity changes of the big square. Taking the median of the stack just leaves us with one image, with no information about the intensity changes of objects in the image. For prediction and analysis in many applications, it is necessary to retain information about when features are changing.



(A) Original 1 (B) Original 8 (C) Original 9

(D) Noisy 1 (E) Noisy 8 (F) Noisy 9

(G) DD-SRAD-RSS-W 1 (H) DD-SRAD-RSS-W 8 (I) DD-SRAD-RSS-W 9

(J) Median

FIGURE 4.5: Showing change preservation - displaying the log of the image

# Chapter 5

# Analysis

## 5.1 Complexity Analysis

Big O notation is used in computer science to quantify an upper bound on algorithm behavior. The "O" corresponds to the "order" of the function being analyzed. Big O notation is stated as a function of the size of input (*N*.) We use Big O to analyze the limiting behavior of a function as the size of an input increases to infinity. Big O is used to delineate "worst-case" complexity of algorithms (*Big-O Cheat Sheet*).

The implementations of these algorithms are in no way optimal - but they were both written by the same individual, so the complexity of both algorithms is comparable. We will not discuss the complexity of each step in the algorithm, but will instead show and discuss the rate-limiting steps for each algorithm.

## 5.2 Complexity of DespecKS

The DespecKS algorithm takes an input image stack of size $N$ x $N$ x $K$ pixels, where $N$ is a measure of input size that we will push towards infinity. The first rate-limiting step in the DespecKS algorithm is to iterate through all the rows, and then all the columns of the image stack, using nested for-loops. Looping through the rows is of complexity O($N$). Subsequently looping through the columns is of complexity O($N$), so we multiply the two complexities together to get O($N^2$).

While looping through columns, this algorithm then selects a window. This window is of size $M$ x $L$. We iterate through all the rows and columns of this window, giving us a complexity of O($ML$). $M$ and $L$ represent 15 x 21, respectively. The nested for-loops iterating through a window are part of the for-loops iterating through the image, so we multiply the complexities to get O($N^2ML$).

Within the window, a KS test is performed between the center pixel of the window and every other pixel in the window. The complexity of a two-way KS test is O($K$). This is because before performing a two-way KS test, one must build two CDFs of the data vectors being tested. When building CDFs, a histograms of intensities must be built, and the frequencies of certain intensities must be summed. This is the rate-limiting step of a two-way KS test. A single sum operation over an intensity vector is of complexity O($K$), because K is the length of the stack (and thus, the intensity vector.) There are two sum operations performed, so the complexity of a two-way KS test is 2O($K$). We drop constants when analyzing complexity, so the simplified two-way KS test is of complexity O($K$).

Multiplying the complexity of the KS test with the previous complexity gives us complexity O($N^2MLK$).

## 5.3 Complexity of DD-SRAD-KS

We compare the DespecKS algorithm to Distance-Driven SRAD using KS distance, as they are the most comparable methods. They both make use of temporal information, and they both perform numerous two-way KS tests. DD-SRAD-KS first takes an input image stack of dimensions $N$ x $N$ x $K$ pixels. Smoothing is done for a number of iterations, which we will call $I$. In our experiments, $I = 200$. For each iteration, we will perform smoothing on all of the images. This gives us complexity of O($I$).

Within each iteration, we will iterate through all the rows and columns of the image stack. This set of nested for-loops, as in the case of DespecKS, will give a complexity of O($N^2$). We will multiply this with O($I$) from the outer loop to get a complexity of O($N^2I$).

The next step is to perform a two-way KS test between the current pixel and all neighboring pixels, within the nested for-loops of the previous step. There are four neighbors for each pixel (north, South, East, and West), and the complexity of each KS test is O($K$), as discussed in the previous section. So there are four KS tests done per pixel, giving us complexity 4O($K$). Constants are dropped when simplifying complexity, so the complexity of this step is simply O($K$). Multiplying O($K$) by O($N^2I$) gives us a total complexity of O($N^2IK$).

## 5.4 Comparison

The two complexities shown above have a couple of common terms: O($N^2K$). So we compare the terms that are different after removing like terms: O($ML$) versus O($I$). For our experiments we have used 200 iterations. We would not prefer to use more iterations as the image may become oversmoothed. The preferred window size for the DespecKS algorithm is 15 x 21, so $M = 15$ and $L = 21$. Thus, $ML = 315$, which leads to significantly increased computational cost (in this case). So, in addition to demonstrating superior performance in terms of solution quality, the DD-SRAD-KS method is also capable of reduced computational cost.

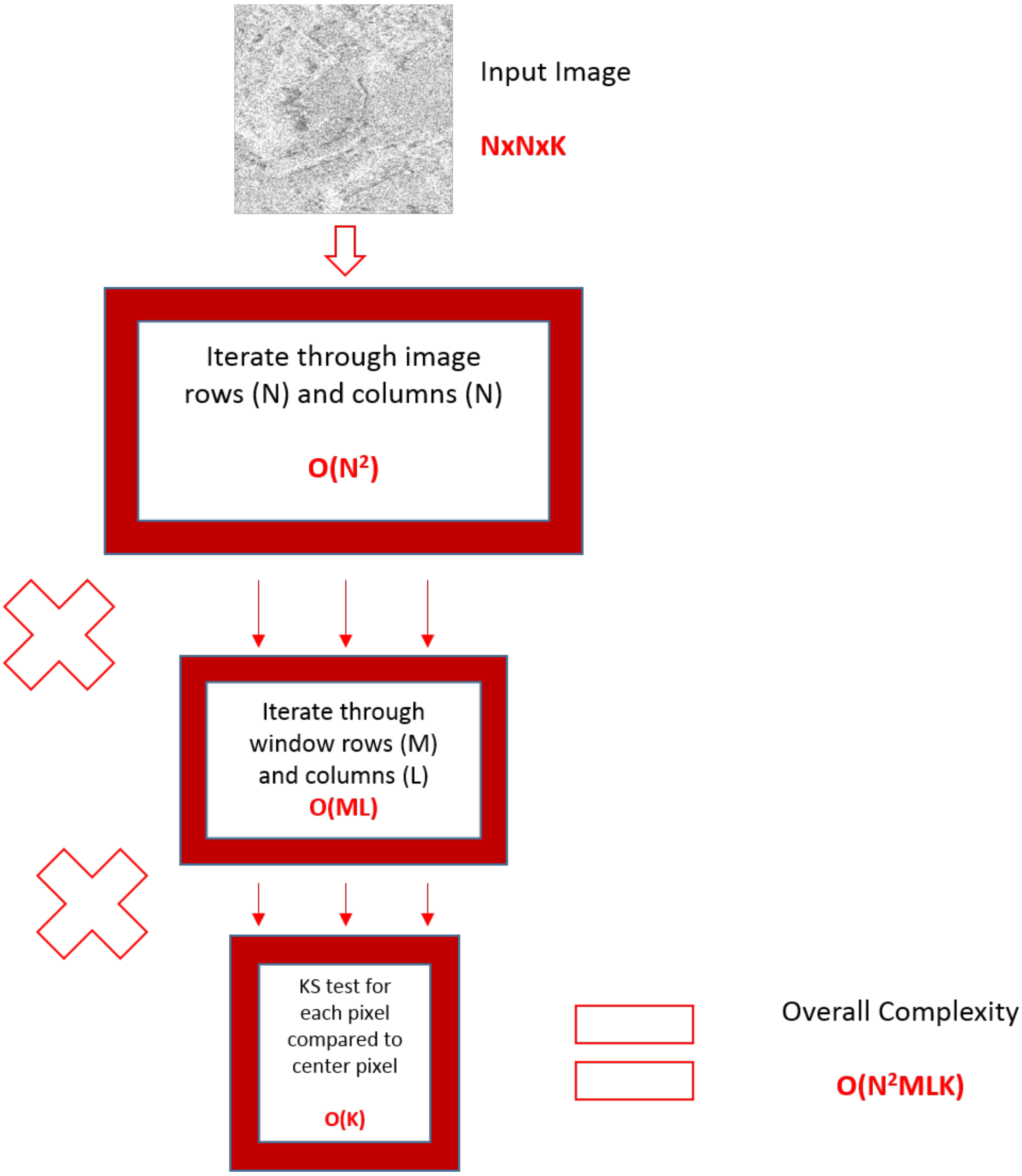

Input Image
NxNxK
Iterate through image rows (N) and columns (N)
O(N²)
Iterate through window rows (M) and columns (L)
O(ML)
KS test for each pixel compared to center pixel
O(K)
Overall Complexity
O(N²MLK)

FIGURE 5.1: Flow diagram of DespecKS algorithm complexity.

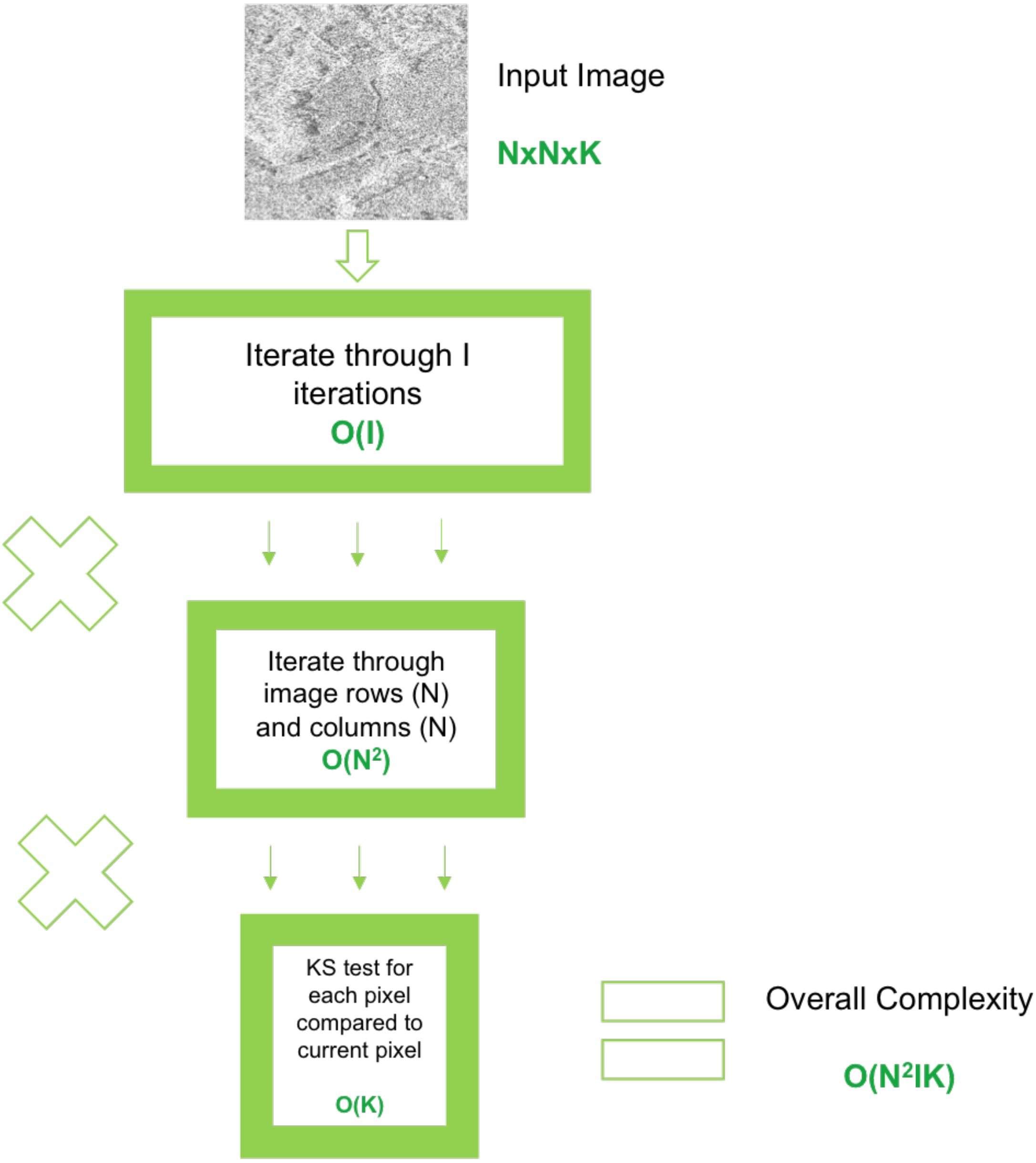

Input Image
NxNxK
Iterate through I iterations
O(I)
Iterate through image rows (N) and columns (N)
O(N²)
KS test for each pixel compared to current pixel
O(K)
Overall Complexity
O(N²IK)

FIGURE 5.2: Flow diagram of SRAD Vector algorithm complexity.

# Chapter 6

# Conclusion

SAR images can be used for many different applications, from farm and ecological analysis, to transportation infrastructure monitoring. The database we draw our images from contains images taken over Virginia, to remotely sense problems with Virginia roads, highways, and bridges. Being able to accurately sense changes in infrastructure topography can lead to better predictions of events, and can support proactive behavior and preventing disastrous consequences of said events (landslides, sinkholes, bridge collapse, etc.) However, SAR images inherently contain speckle noise, which degrades image structure and makes accurate analysis of amplitude data difficult. To ensure accuracy of predictions, so that false alarms are not raised, we must denoise SAR images before performing any type of analysis with them.

For denoising SAR images, we have developed a method that performs better than leading methods in speckled image denoising. Our method, DD-SRAD, performs better than many single image denoising methods. DD-SRAD also has lower MSE and higher PSNR than the only other popular denoising method for SAR image stacks (that utilizes temporal information.) DD-SRAD not only removes more noise than this method, DespecKS, but also preserves more temporal uniqueness than DespecKS. DD-SRAD also has lower overall complexity than DespecKS, even though both methods make use of the temporal nature of the image stack.

To continue this work, scaled distances can be implemented, as well as testing different types of distances besides the three tested here (RSS, KS, and Bhattacharyya.) Also, weighted distances can be implemented for KS and Bhattacharyya distances.

Moving away from diffusion, an improvement can be made to the DespecKS algorithm to lower the complexity and improve the performance. Instead of windowing through the whole image to find SHP, homogeneous regions can be found using our homogeneous region detection method. Each homogeneous region can be averaged. Then, SHP analysis can be performed on the remaining non-homogeneous edges (roads, etc.) We will implement this improvement on DespecKS in the near future, and compare the results to DD-SRAD.

# Bibliography

A. Ferretti A. Fumagalli, F. Novali C. Prati F. Rocca and A. Rucci (2011). “A New Algorithm for Processing Interferometric Data-Stacks: SqueeSAR”. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.9, pp. 3460–3470.

Bhattacharyya, A. (1943). “On a measure of divergence between two statistical populations defined by their probability distributions”. In: *Bulletin of the Calcutta Mathematical Society* 35, pp. 99–109.

D. T. Kuan A. A. Sawchuk, T. C. Strand and P. Chavel (1987). “Adaptive restoration of images with speckle”. In: *IEEE Trans. Acoust., Speech, Signal Processing* ASSP.35, 373–383.

Hogema, J. (2005). “Shot group statistics”. In:

Immerkaer, J. (1996). “Fast Noise Variance Estimation”. In: *Computer Vision and Image Understanding* 64.2, pp. 300–302.

*Interstate 35W Bridge Collapse Critical Response*. Online. URL: http://www.govtogovsolutions.org/Default.aspx?tabid=132.

Jarque, C. M. and A. K. Bera (1987). “A Test for Normality of Observations and Regression Residuals”. In: *International Statistical Review* 55.2, pp. 163–172.

Ju, C. and C. Moloney (1997). “An edge-enhanced modified Lee filter for the smoothing of SAR image speckle noise”. In: *International Workshop on Acoustic Signal Enhancement*.

Lee, J. S. (1980). “Digital image enhancement and noise filtering by using local statistics”. In: *IEEE Trans. Pattern Anal. Machine Intell.* PAM1.2.

Liew, S.C. (2001a). *Interpreting SAR Images*. Online. URL: http://www.crisp.nus.edu.sg/~research/tutorial/sar_int.htm.

— (2001b). *Microwave Remote Sensing*. Online. URL: http://www.crisp.nus.edu.sg/~research/tutorial/mw.htm.

Massey, F. J. (1951). “The Kolmogorov-Smirnov Test for Goodness of Fit”. In: *Journal of the American Statistical Association* 46.253, pp. 68–78.

McCandless, S.W. and C.R. Jackson. *SAR Marine User’s Manual*. NOAA.

Perona, P. and J. Malik (1987). “Scale-space and edge detection using anisotropic diffusion”. In: *Proceedings of IEEE Computer Society Workshop on Computer Vision*, pp. 16–22.

Rowell, E. *Big-O Cheat Sheet*. Online. URL: http://bigocheatsheet.com/.

*SAR Imagery* (2016). Online. URL: http://treuropa.com/technique/sar-imagery/.

Wolff, Christian. *Radar Basics*. Online. URL: http://www.radartutorial.eu/20.airborne/ab07.en.html.

Yu, Y. and S. T. Acton (2002). “Speckle Reducing Anisotropic Diffusion”. In: *IEEE Transactions on Image Processing* 11.11, pp. 1260–1270.