Design and Development of a Java-based Distributed System Tool for Synthetic Aperture Radar Image Analysis

By
Carlos Huallparimachi Suarez

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
COMPUTER ENGINEERING
(Software Engineering)

UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
2004

Approved by:

_____	_____
Néstor Rodríguez, PhD Member, Graduate Committee	Date
_____	_____
Manuel Rodríguez, PhD Member, Graduate Committee	Date
_____	_____
Domingo Rodríguez, PhD President, Graduate Committee	Date
_____	_____
Miguel A. Pando PhD Representative of Graduate Studies	Date
_____	_____
Jorge Ortiz, PhD Chairperson of the Department	Date

Abstract

Image processing is used to analyze a digital image or to transform it into a new image. Different techniques for the manipulation, correction, and enhancement of digital images have been used for years. One of the main objectives of these techniques has been the removal of defects from images obtained through a wide variety of equipments. In this work we describe the design and development of a Java-based distributed system tool-environment for image analysis with special attention given to synthetic aperture radar (SAR) imaging applications. This tool-environment allows a given user to effect important image processing functions such as to visualize, manipulate, improve, filter, detect edges, and reduce the noise on SAR images. Also, this system has the unique option of allowing end-users to add their own customized algorithms as encapsulated operators to act on elements resident on local or remote SAR images-servers on a computer network.

Resumen

El procesamiento de imágenes es utilizado para analizar una imagen digital o para transformarlo esta en una nueva imagen. Durante años han sido utilizadas diferentes técnicas para la manipulación y corrección de imágenes digitales. Uno de los principales objetivos de estas técnicas es remover los defectos de las imágenes que son adquiridos a través de una amplia variedad de equipos. En este trabajo describimos el diseño y desarrollo de una herramienta desarrollada en Java para el análisis de imágenes en un ambiente distribuido con especial énfasis en las imágenes obtenidas por los sistemas de Radar de Apertura Sintética (SAR, por sus siglas en ingles). Esta herramienta permite a los usuarios efectuar importantes funciones dentro del procesamiento de imágenes, tales como, visualización, manipulación, mejoramiento, filtrado, detección de bordes, reducción de ruido en imágenes SAR. También este sistema tiene una opción que permite a los usuarios finales agregar los algoritmos diseñados por ellos mismos, como operadores encapsulados que actúan sobre las imágenes que se encuentran localmente o remotamente en los servidores de imágenes SAR en una red de computadoras.

To God, to my family, and best friends for their help.

Acknowledgement

I would like to thank my advisor Dr. Domingo Rodríguez for giving me the opportunity to work with him, for his support, patience, dedication and help in this thesis. I would also like to thankful to my graduate committee members: Dr. Néstor Rodríguez, and Dr. Manuel Rodríguez, for their mentoring and corrections that contributed to the enhancement of this thesis.

My thanks to my family for their love and support. Special thanks to Maria Diaz Figueroa, who encourage me and help me in many aspects. Also, I would to express my sincere appreciation for all my friends that contributed to the successful development of this thesis.

Finally, I want to thank the Department of Electrical and Computer Engineering at University of Puerto Rico-Mayagüez Campus for this opportunity to follow graduate studies in this prestigious university and the Precise Project during my graduate studies. I want to apologize for any merits omission, it is not intentional.

Table of Contents

List of Figures ..... **ix**

List of Tables ..... **xi**

Chapter 1

Introduction **1**

1.1 Problem Formulation 3

1.2 Justification 4

1.3 Research Objectives 6

1.4 Research Methodology 6

1.5 Original Contributions 7

1.6 Structure of this Thesis 8

Chapter 2

Survey of Related Work **9**

2.1 Commercial Products 9

2.2 Non Commercial Products 10

Chapter 3

Synthetic Aperture Radar System 14

3.1 Introduction 14

3.2 SAR Imaging System 15

3.3 Properties of SAR 18

Chapter 4

Fundamental Concept in Image Processing 19

4.1 Digital Images 19

4.2 Image Data 20

4.3 Image Processing 21

4.3.1 Image Enhancement 22

4.4 Mathematical Operations 22

4.4.1 The Fourier Transform 22

4.4.2 The Discrete Fourier Transform 23

4.4.3 Convolution 24

4.4.4 Correlation 24

4.5 Smoothing Operators 25

4.5.1 Linear Filter 25

4.5.2 Non-Linear Filter 27

4.6 Edge Detection 28

4.6.1 Sobel Edge 28

4.6.2 Roberts Edge 29

4.6.3 Prewitt Edge 29

4.6.4 Frei and Chen Edge 30

Chapter 5

Overview of the JSIM Prototype System 31

5.1 JSIM Prototype 31

5.2 JSIM Network Architecture 33

5.3 Operator Agent Environment 35

5.4 History List and Operator Encapsulation 37

5.5 New Operators Customized 39

5.6 Template class operator design 41

5.7 JSIM Servlet Model 43

5.8 Metadata 44

5.9 Comparison between JSIM and ImageJ 45

Chapter 6

Usability Interface Study 46

6.1 Introduction 46

6.2. Objectives 46

6.2 User Profile 47

6.3 Test Methodology 47

6.3.1 Introduction. 47

6.3.2 Training 48

6.3.3 List of Tasks 48

6.3.4 Questionnaire 48

6.3.5 Interview 49

6.4 Discussion and Results 49

6.4.1 Data Analysis 49

6.4.2 Result Evaluation 51

6.4.2.1 Overall Reactions 53

Chapter 7

Conclusions and Future Work 57

7.1 Conclusions 57

7.2 Future Work 59

References 60

Appendix A – Instruction Test 65

Appendix B – Questionnaire Background Information 69

Appendix C – Final Questionnaire User Satisfaction 70

Appendix D – Application Form Data Collection 74

Appendix E – JSIM Presentation 75

List of Figures

Figure 1 Scenario of a distributed system to acquire terrestrial images 3

Figure 2 Example of SAR Processing 16

Figure 3 SAR Raw Data Generation..... 17

Figure 4 Image Formation Algorithms 17

Figure 5 Coordinate systems for an image 20

Figure 6 Components of a digital image..... 21

Figure 7 Graphics User Interface of JSIM..... 32

Figure 8 JSIM Architecture 33

Figure 9 Internal description of the architecture..... 34

Figure 10 Distributed System Activities..... 35

Figure 11 Operator Agent Environments..... 36

Figure 12 Set of Operators to Encapsulate 37

Figure 13 Encapsulated Operators..... 38

Figure 14 Interface history lists 38

Figure 15 Interface List Operators..... 39

Figure 16 Methodology Pattern for Operator Addition. 40

Figure 17 Template class to create news operators 41

Figure 18 Interface to create new operators..... 42

Figure 19 JSIM Using Servlets..... 43

Figure 20 Sections of the experiment 47

Figure 21 Total run times to complete all task by each participants 51

Figure 22 Overall Reactions about JSIM application 53

Figure 23 Learning with JSIM application 54

List of Tables

Table 1 Low pass filter 3x3 26

Table 2 Example Low pass filter 3x3 26

Table 3 High pass filter 3x3 27

Table 4 Laplacian filter 3x3 27

Table 5 Kernels to Sobel Edge 29

Table 6 Kernels to Roberts Edge 29

Table 7 Kernels to Prewitt Edge 29

Table 8 Kernels to Frei and Chen Edge 30

Table 9 Features of JSIM and ImageJ 45

Table 10 Run time of each task by the participants 50

Table 11 Questionnaire results by each participant 52

Chapter 1

Introduction

Image processing involves the manipulation of the pixels of one digital image. Various operations may be performed on the pixels of the original image to produce the new image. The purpose of image processing is to improve the visual appearance of images obtained through a wide variety of equipments, or to prepare images for measurement of features and structures present. In general, these images can be obtained from a diverse set of instruments such as a low resolution digital camera to an expensive high resolution medical imaging scanner, or by sensors mounted on satellites or airplanes. Regardless of how an image is acquired, image processing applications typically provide three basic functions for handling images: loading, rendering, and manipulating [1].

The processing and analysis of images require a great amount of computational power. At the present time it is very important to develop tools that allow us to make our work with image processing faster and more efficient. Also, large amount of diverse data is now available, such as very high resolution images and images from spectrometers.

Some expert and data fusion systems offer a facility to integrate various data through various approaches and techniques. The results of such approaches allow access to diverse data (vegetation, disturbed areas, burned zones, mixed zones [2][3].

The internet has been the breeding ground for many exciting new technologies in the recent years; one of the most important of these technologies is the Java programming language [4]. Java is a language that seems particularly well-suited for the development of software in network environments. Within a short period of time, server-side computing has already embraced Java. Even though the imaging industry has been slow to react to Java, many industry leaders have realized its merits and are moving towards it.

The World Wide Web (www) technology provides the medium through which millions of images are moved daily between computers at all points on earth. The image processing plays a very important role in many high-tech areas, including medical imaging, satellite imaging, and astronomy, among others. The processing and analysis of images have several applications in many different areas of science and engineering, among these areas we can mention: Defense and Intelligence, Geospatial Data Processing, Document Image Processing, Bi-Informatics, Applied Imaging Research, Digital Photography, and Retail Teaching [4].

Many commercial products are available to support image processing specialists. These products have many features for loading, manipulating, and rendering images. Companies often conduct research works in image processing to develop new

technologies that they commercialize in the software market. The resulting software products tend to be rather expensive for academic settings worldwide, or are usually not widely accessible to most interested users throughout the developing world. This work seeks to contribute to the free availability of image analysis tools for academic purposes. Figure 1 shows a scenario of a distributed system to acquire terrestrial images from diverse sensors.

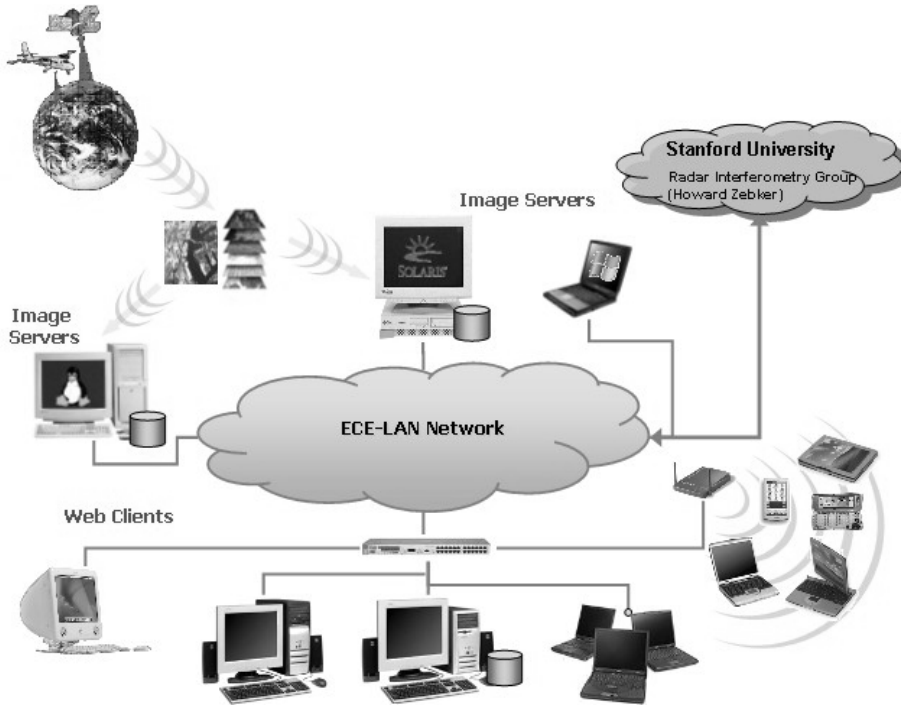


Figure 1 Scenario of a distributed system to acquire terrestrial images

1.1 Problem Formulation

With current advanced data acquisition and imaging technologies, it is now possible to acquire SAR images in many different ways, from sensors mounted on satellites and/or

airplanes platforms. In the recent years there has been a large production of SAR images, which are used by diverse studies in different areas of science and engineering. The SAR technology allows obtaining terrestrial images of high spatial resolution, even through cloudy skies, to help us improve the comprehension of environmental phenomena in the geosciences. It is necessary to have modern automated tools to manipulate these images and develop new applications.

This stated necessity generates a justifiable motivation for the development of a new set of java-based image analysis tools that contribute in an integrated manner to the difficult problem of distributed automated information processing. The researchers and/or students need tools that can be used in their computers or through the Internet. These applications must process the image data available at Internet in an efficient manner, offering modularity, scalability, and robustness. Other special feature might be the availability of operator agents (algorithms), which could be developed for the users of the tool environment in order to conduct studies over several types of images.

1.2 Justification

Considering advances in information technologies, we wish to offer an open-source application, platform independent for analyzing and manipulating SAR images. The purpose of this research is to design and develop an application for the analysis of SAR images. This tool provides friendly user interfaces in which is possible to view and analyze SAR images using basic, advanced and user-defined image operators.

The application could be installed in a portable computer or work station. It has two modes of operation: client-server mode where the operators selected can be over different servers and local mode where the operators are installed in the machine.

Creation of new operators is possible from existent operators which are defined as default. The new operators are encapsulated as a sequential command. Another important option is that users can write their own algorithms to analyze and display data. The advantage and value added of this option is the economy of time, when programming new operators.

For users on the Internet, we will design and develop an application with Java-servlets technology to analyze and manipulate SAR images. A relevant advantage is that the users are able to use the tool environment from any part of the world across of the Internet.

Other main component of this tool environment is integration of metadata with each SAR images. This metadata has its own respective descriptive information of the images, like the place of the image, date, resolution, latitude, longitude and other information related. The end users could manipulate metadata if they so desire.

Finally, to adhere to the philosophy of object-oriented, the application is implemented as a collection of re-usable components and the public features of each component are fully described.

1.3 Research Objectives

The main goal of this research is to design and develop a Java-based distributed system tools prototype for image analysis with special attention given to SAR images, this tool allows us to visualize, manipulate, improve, filter, detect the edges and reduce the noise on SAR images.

This research also hopes to provide friendly user interfaces and an extensible capability to java-based image analysis tools for operator extension and manipulation. The creation of new operators based on existing operators, encapsulating commands in a sequential order. Also, the tool allows writing new operator agents (algorithms), implementing new java classes that should be integrated to the existing tool. The tool allows adding metadata with each SAR image. This metadata is the description of the attributes of the image.

Allow the java-based image analysis tool environment a degree of portability permitted in a given workstation network configuration. Design the tool to run on various platforms and to be used through the Internet.

1.4 Research Methodology

The following steps will be performed to accomplish the objectives of the proposed research work:

- Literature research about fundamental principles involved in image analysis, image processing, convolution, correlation and, FFT. This stage includes analysis of theoretical information and review of existing software packages for image analysis for a better understanding of the current state of knowledge.
- Analyze the requirements and features of the system in which we define the operators that the tool environment has by default.
- Design the Graphics User Interface (GUI) for the application.
- Implement a prototype using standard distributed system environment (Java RMI) and object-oriented approach. This prototype will help us evaluate the proposed model and detect potential deficiencies.
- Develop a web page with Java Servlets for image analysis where users can test the tool environment and the operators remotely through Internet.
- Implement a Web page with the information about JSIM, which have the description, features, documentation, downloads and references.
- Conduct a usability evaluation of prototype, with a sample of users. The feedback from the users will be important to improve the tool.

1.5 Original Contributions

This research expects to make the following contributions.

- Contribute to the free availability of image analysis tools for academic purposes.
- Developed a Java-based tool that allows to users to manipulate and analyze SAR images.

- Create new customized algorithm as encapsulated operators.
- Allow to add metadata with each SAR image.

1.6 Structure of this Thesis

The remainder of this thesis is organized as follows. Chapter 2 Survey of Related Work, Chapter 3 provides a general introduction about Synthetic Aperture Radar System. Chapter 4 provides Fundamental Concept in Image Processing. Chapter 5 provides a step by step description of the developed environment, named JSIM. We describe the prototype, the network architecture behind it, its special features and how it relates to its special operators, template class, servlets model, and metadata. Chapter 6, presents usability test plan, as well as results obtained and their analysis. Chapter 7 presents the final conclusions and future work. Finally, the References and the respective Appendixes are presented.

Chapter 2

Survey of Related Work

2.1 Commercial Products

Many commercial products are available to support image processing. These products have many features for loading, manipulating, and rendering, the images. Below, we mention some of these products with its respective features.

The company Research System Inc provides the Environment for Visualization Images (ENVI) [5], this software combines a complete image processing package with multiple applications for diverse areas in science and engineering. ENVI provides geometric correction, terrain analysis, radar analysis, raster and vector GIS capabilities, extensive support for images from a wide variety of sources. The radar toolset helps you explore your SAR data and identify features.

The Environmental Systems Research Institute (ESRI) is another company with a range of products to support image processing and other tools for manipulating image data. They develop a variety of applications to work with Geographical Information

System (GIS). The GIS has the ability to organize information into a series of layers that can be integrated using geographic location. These include alphanumeric data and image data (e.g., point, line, polygon, raster image) [6].

PCI Geomatics provides integrated software for remote sensing, image processing, GIS/Spatial Analysis, Cartography and others. There is a module that allows to the users work with RADARSAT image [7].

Khoros is an integrated development environment that allows researchers and scientist to solve problems related to scientific computing and visualization. Khoros contains over 300 programs, or operators for information processing, data exploration, image and signal processing, and data visualization. Generalized to form a broad-level technology, these operators facilitate problem solving in a wide variety of application domains used in research, science, government and industry [8].

These companies make research in image processing to obtain new technologies that they commercialized in the software market. The results are expensive products that are not accessible to everybody working in an Image Processing related field.

2.2 Non Commercial Products

We present some of the work and researches most relevant to the work made at the universities and diverse studies presented in conferences related to image processing.

Don Murray, Bill Hibbard, Tom Whittaker, and James Kelly [9] have developed a tool for visualizing and analyzing remotely sensed data, and examples of image display and analysis tools. They used library VisAD for their implementation. VisAD is an open-source Java library for building interactive and collaborative visualization and analysis tools. The main feature of VisAD is a mathematical data model that provides transparent access to data independent of storage format and location.

Seng Chuan Tay and Kim Hwa Lim [10] have developed an application for Internet, Web-based processing for remotely sensed data, they adopt the client-server processing technology to process the NOAA data in a distributed manner, thereby reducing the program elapsed time. They used Java socket and the TCP/IP protocol. Their experiments show that web-based processing is a viable approach to reduce the time required for hotspots detection.

The Unidata is a diverse community of education and research institutions vested in the common goal of sharing data, tools to access the data, software tools to use and visualize the data, and resources. The Unidata Program Center offers software and services that enable universities to acquire and use atmospheric and related data on their own computers, usually in real time. As the world in moving to a new age of interconnected computer systems, new applications will be required to facilitate the education and research. Unidata's MetApps is a project to develop Java applications for analyzing and visualizing meteorological and related geosciences data, including remotely sensed data such as satellite and radar data [11].

The distributed systems are a set of solutions to integrate the systems and share data, and increase the computational power. For example the Italian Scientific Community is working in Earth Observation from the Space (EOS) sector deals with voluminous amount of data acquired and/or generated by various research centers and institutes. They are migrating to a distributed system where the data and metadata is being sheared in a dynamical and transparent way among other advantages [12].

G. Walter, F. Warmendam, and P. Farris-Manning [13] have developed an open source tool for geospatial image exploitation; this tool is free and multi-platform. The viewing supports basic display and image manipulation, also an extensible command shell for more specialized operations. The command shell provides access to a powerful programming and scripting language (Python) which supports custom extensions and research tool development.

Do-Hyun Kim and Min Soo Kim [14] have designed and implemented the open web GIS service system based on the web GIS architecture of Open GIS Consortium (OGC); this system provides the inter-operatibility, open environment, and spatial data using to transfer a format based on XML.

ImageJ provides full features image processing program developed at the NIH. ImageJ is used on a routine by biologists worldwide to assist them with the processing and analysis their images. ImageJ also has an extensive library of plug-ins developed by users [15].

The Java Vision ToolKit (JVT) is a Java-based software library for machine vision and image processing applications. It is based on the Java Advanced Imaging library, adding machine vision algorithms for 2D and 3D images to this framework [16].

Large amount of data is now available, such as very high resolution images and images from spectrometers. Expert system offers a facility to integrate various data. The results of such approaches allow quick access to diverse data (vegetation, disturbed areas, burned zones, mixed zones) [17].

In this area of image processing exist many private companies how the mentions previously, government as NASA, universities around the world, Scientists; dedicated in the research of new algorithms and techniques for manipulation, correction and enhancement of digital images.

The broad field of image processing involves the manipulation and analysis of visual data; fundamental mathematical procedures (known as algorithms) are applied to the data, enabling it to be processed by computers or hardware designed for this purpose. New algorithms are constantly being developed for image encoding, compression, feature extraction, image analysis, and spatial filtering.

Chapter 3

Synthetic Aperture Radar System

3.1 Introduction

Synthetic aperture radar (SAR) are sensor-based system mounted on space borne or airborne platforms which have as main function the active, low-power, microwave illumination of the Earth's surface in order to obtain imaging information for geosciences applications. Its active sensor system nature allows SAR to work all the time, day and night, under any climatic condition and to obtain images of higher spatial resolution than those generated with conventional instruments [18][19]. These characteristics make SAR a very attractive system in remote sensing applications.

The synthetic aperture radar technology, in general, has been one of the most exciting and progressive fields in remote sensing in the last years. The SAR images generated by the SAR systems can be used to solve many problems in land and oceanographic remote sensing [20][21]. The SAR technology has resulted in marked improvements in spatial resolution imaging operations when observing a ground scene from aircrafts or satellites and it can be used to estimate also parameters and features such as the dampness

of soils, wetlands, fluvial structures, metallic surface structures, thickness of the forests, or the roughness of the sea.

Processing of SAR data is required to extract relevant features, such as objects or buildings. Detection of such objects is based on the detection of locally bright pixels, followed by clustering of neighborhoods of pixels [22][23].

3.2 SAR Imaging System

A radar system illuminates an area with microwaves, and records the strength and travel-time of the returned signals. This allows the range (or distance) of the reflecting objects to be determined [24].

If the radar is attached to a moving platform, either a satellite or an aircraft, then it is possible to combine reflected signals from along the flight path to synthesize a very long antenna. The aperture, or area used to receive signals, is created artificially during the signal processing.

The synthetic aperture gives the radar a high resolution in the azimuth (or along-track) direction - the line of the flight path. The resolution in the range direction is determined by the duration of the transmitted pulses. In practice, to achieve a fine resolution, the pulse width would be too short to contain sufficient power. Therefore, longer, frequency-modulated pulses (linear chirps) and transmitted. This complicates the

image formation process. However, with the use of parallel computing, SAR images can be formed in real-time [24]. The next Figure 2 shows an example of SAR operation and processing.

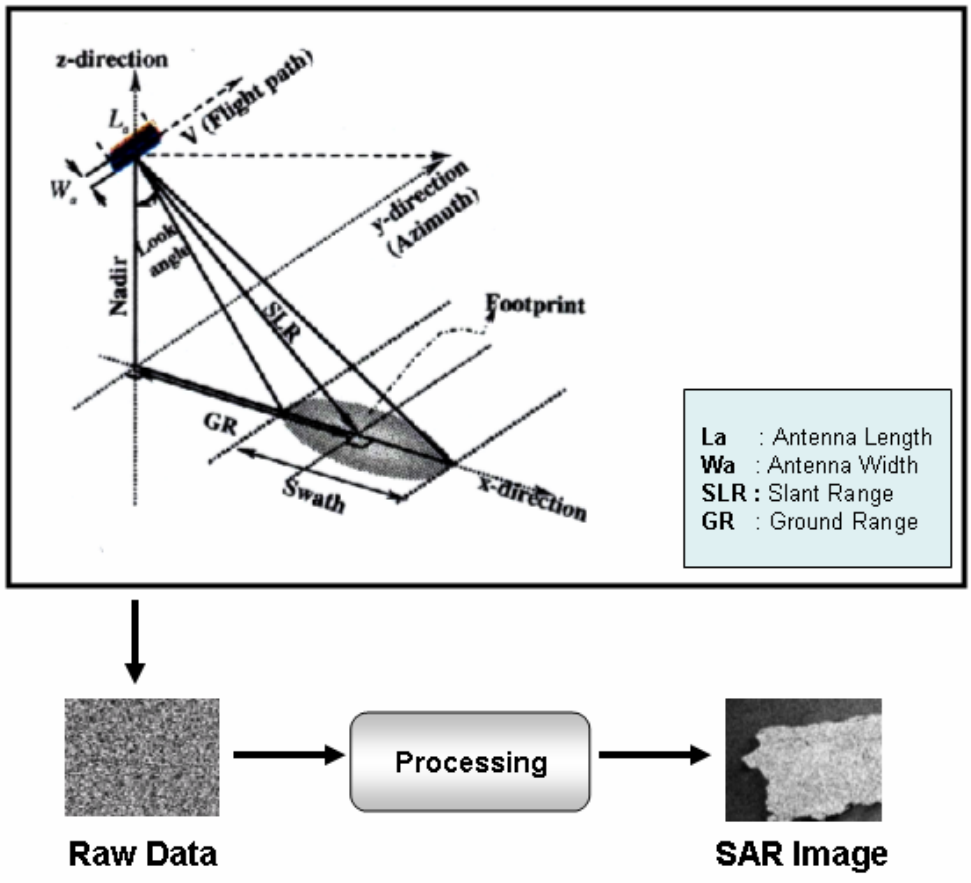


Figure 2 Example of SAR Processing

After the sensor illuminates the target area, an image is generated; this image is called raw image because it is not processed completely. SAR imagery goal consists of recovering the original image or reflectivity function from raw data as shown Figure 2. Two steps are necessary to recover the image.

In the first step, the raw data is generated using time-frequency processing techniques such as the ambiguity function, as illustrated in Figure 3. The process consists of computing the correlation between the transmitted and the reflected signal, which corresponds to the spectral density reflected by the target.

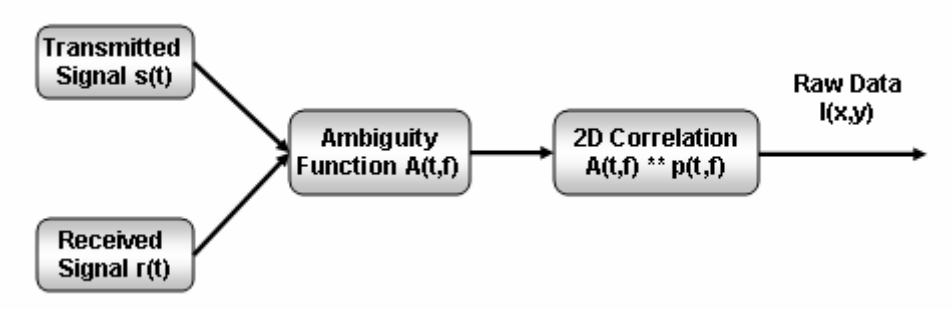


Figure 3 SAR Raw Data Generation

In the second step the image is recovered from raw data. The algorithm shown in Figure 4, was proposed by Franceschetti. The algorithm performs cyclic correlation operations between raw data and the scene spectral density.

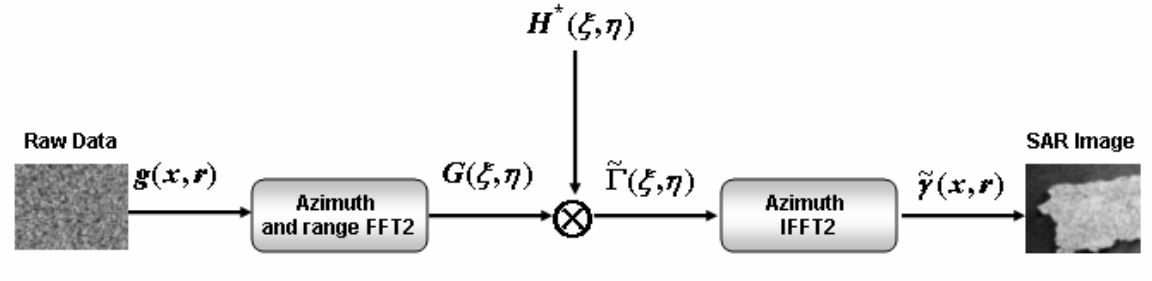


Figure 4 Image Formation Algorithms

3.3 Properties of SAR

Advantages

- A SAR is an active instrument - it transmits its own radiation. This means it is not dependent on the sun and can operate continuously, day and night.
- Microwave radiation is unaffected by cloud cover, fog, or precipitation so it can image the Earth's surface in all weather conditions.
- SAR observes structures at length scales other than those captured by optical instruments. The two datasets are complementary and record different properties.
- SAR is capable of continuous observation of dynamic phenomena such as the monitoring of ocean currents and sea ice, and changes in water and vegetation coverage.

Disadvantages

- At the resolutions typical of easily available SAR datasets, there is little redundant information: objects like trees, vehicles and buildings typically occupy only one or two pixels.
- SAR is highly dependent on maintaining precise time relationships between the transmitted and received signals. All such coherent imaging technologies suffer from the problem of speckle.

Chapter 4

Fundamental Concept in Image Processing

This chapter deals with the basic concepts related with the treatment of image processing. Design and development of image processing algorithms require a previous knowledge about characterization, classification, representation, operations, methods for image processing, and their applications. This chapter provides the reader with a review of fundamentals concepts related with image processing.

4.1 Digital Images

Images are produced by a variety of physical devices, such as x-ray devices, microscopes, radar and ultrasound, and used for a variety of purposes, including entertainment, medical, business, industrial, military, security and scientific. The goal in each case is for an observer, human or machine, to extract useful information about the scene being imaged.

An image is a representation of two dimensions of the visual world. An image is a function of two dimensions f (x, y). For monochrome images, the value of the function at any pair of coordinates, x and y, is the intensity light detected at that point. In the case

of color images, f(x, y) is a vector-valued function. The function f(x, y) must be translated into a rectangular array of numerical data. This digital representation is an approximation of the original image to manipulate the image using a computer. Translation of f(x, y) into an appropriate numerical form is accomplished by the processes of sampling and quantization [**26**]. The image is defined in a coordinate system whose origin is defined as the upper-left corner of the image (Figure 5).

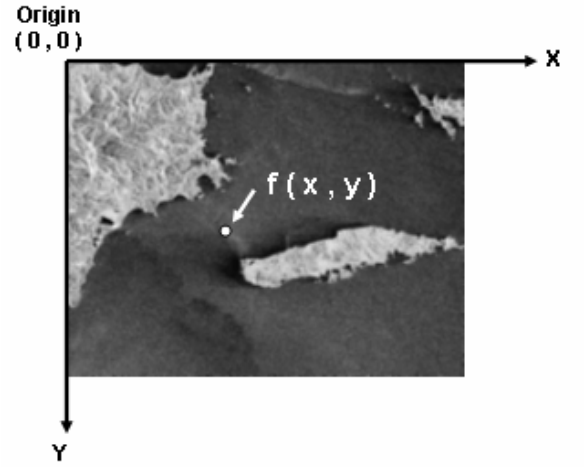


Figure 5 Coordinate systems for an image

An image is divided into N rows and M columns. The intersection of a row and a column is called a pixel. Each pixel has an (x, y) coordinate that corresponds to its location within the image.

4.2 Image Data

Image data is, conceptually, a three-dimensional array of pixels. Each array is called a band. The number of rows specifies the image height of a band, and the number of columns specifies the image width of a band.

Monochrome images, such as a grayscale image, have only one band. Color images have three or more bands, although a band does not necessarily have to represent color. The satellite images of the earth may be acquired in several different spectral bands, such as red, green, blue, and infrared. Figure 6 shows the components of a digital image.

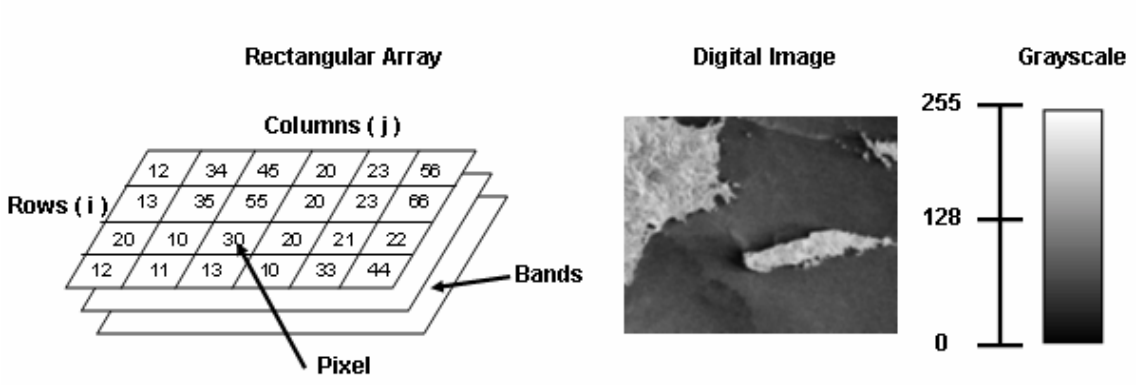


Figure 6 Components of a digital image

4.3 Image Processing

Image Processing is a general term for the wide range of techniques that exist for manipulating and modifying images in various ways. Some of the most fundamental manipulations are image formation, image enhancement and restoration, noise removal, geometrics transformations, features extraction, edge detection, and storage transmission [**27**].

The acquired image usually suffers from additive noise, linear distortions, nonlinear distortions, and possibly some lost samples. The image restoration process involves the removal of the noise and the distortions. It also involves filtering methods and statistical

techniques. Geometrics distortions may require the application of geometrics transformations. The visualization of the restored image can require some enhancement in order to improve the perception of the image. Enhancement techniques include grayscale modification, edge enhancement, and contrast modification [**27**].

4.3.1 Image Enhancement

Image enhancement can be defined as conversion of the image quality to a better and more understandable level for feature extraction or image interpretation. The main goal is to improve the perception of the image through modification of intensity functions. Several different classes of methods are available to enhance the image. For example the grayscale transformation allows changing the perception of the image by a careful remapping of the intensity. Other methods involve modification of the histogram of the image, and finally, the methods based in linear filtering (low and high pass filters) that allow us enhance features of the images.

4.4 Mathematical Operations

4.4.1 The Fourier Transform

The *Fourier transform*, in essence, decomposes or separates a waveform or function into sinusoids of different frequency which sum to the original waveform. It identifies or distinguishes the different frequency sinusoids and their respective amplitudes. The Fourier transform of f(x) is defined as:

$$F(u) = \int_{-\infty}^{\infty} f(x) \exp[-2\pi i u x] dx$$

The inverse Fourier transform is:

$$f(x) = \int_{-\infty}^{\infty} F(u) \exp[2\pi i u x] du$$

The Fourier transform can extend easily to two or more dimensions. The two-dimensional Fourier transform of f(x, y) is defined as:

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) \exp[-2\pi i(ux + vy)] dx dy$$

The inverse Fourier transforms in two-dimensional is:

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) \exp[-2\pi i(ux + vy)] du dv$$

4.4.2 The Discrete Fourier Transform

The Discrete Fourier Transform is defined in two dimensions as:

$$F(u,v) = \frac{1}{WH} \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} f(x,y) e^{-2\pi i(ux/W + vy/H)}$$

The inverse is:

$$f(x,y) = \sum_{u=0}^{W-1} \sum_{v=0}^{H-1} F(u,v) e^{2\pi i(ux/W + vy/H)}$$

The reason for the interest in the Fourier transform is the convolution theorem. The convolution theorem state that convolution in the spatial domain is equal to multiplication in the frequency domain (and vice versa).

Another motivation for using the Fourier transform lies in the existence of fast algorithms for its evaluation.

4.4.3 Convolution

In image processing the convolution is an operation taken between two images. One image is typically smaller than the other image. The smaller image is called the kernel of the convolution. The convolution operation consists of a weighted sum of the area surrounding an input pixel. In fact, the convolution is the correlation between the image and a reverse kernel.

The convolution is a primary technique for spatial filtering. A spatial filter takes the local area around a pixel in an image and performs an operation on each of the pixels in order to create a new pixel in an output image. The dimensions of the area about the pixel of interest are given by the convolution mask and form the outline for a convolution window. The convolution window is moved so that it centers about each pixel in the input image. After centering, a new pixel is computed using an arithmetic sum of products.

The discrete convolution is defined in two dimensions as:

$$h(x,y) = f * g = \sum_{u=0}^{W-1}\sum_{v=0}^{H-1} f(u,v)g([x-u],[y-v])$$

4.4.4 Correlation

Correlation is a very efficient tool to match images. It is quite robust to noise, and can be normalized to allow pattern matching independently of scale and offset in the images.

It has serious drawbacks however, in the case images of faint sources. The cross correlation between two functions f and g is defined as:

$$g(x,y) = f * g = \sum_{u=0}^{W-1}\sum_{v=0}^{H-1} h^*(u,v)\, f(x+u, y+v)$$

4.5 Smoothing Operators

These algorithms are applied in order to reduce noise over images for further processing. We distinguish between linear and non- linear algorithms [**27**] [**26**].

4.5.1 Linear Filter

The nature of the filter is determined by our choice of kernel coefficients. Standard kernels are available, with which we can accomplish blurring or sharpening of an image. The kernel can be 1x 1, 3 x 3, 5 x 5, M x N, and so on. A large kernel size affords a more precise filtering operation by increasing the number of neighboring pixels used in the calculation. However, the kernel cannot be bigger in any dimension than the image data.

4.5.1.1 Low Pass Filter

A low pass filter attenuates the high spatial frequency components of an image and has little affect on the low-frequency components. The low pass filter smoothes or blurs the image. This tends to reduce noise, but also obscures fine details. Any convolution kernel with all positive values will act as a low pass filter. Now we presents different low pass filter (Table 1).

Low Pass Filter 1 / 3		
0	1/6	0
1/6	1/3	1/6
0	1/6	0

Low Pass Filter 1 / 4		
1/16	1/8	1/16
1/8	1 / 4	1/8
1/16	1/8	1/16

Table 1 Low pass filter 3x3

The average filter is an example of a low pass filter. The kernels of a low pass filter are coefficients that are positive and the sum is one (Table 2).

Low Pass Filter		
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Low Pass Filter			
1/12	1	1	1
	1	4	1
	1	1	1

Table 2 Example Low pass filter 3x3

4.5.1.2 High Pass Filter

The high pass filter has the opposite effect of the low pass filter, accentuating the high frequency components and offering little affect on the low frequency components. The effect of passing an image through a high pass filter is a sharpen image with increased detail in the areas of brightness transition.

The high pass filter is accomplished using a kernel containing a mixture of positive and negative coefficients (Table 3).

High Pass Filter		
-1	-1	-1
-1	8	-1
-1	-1	-1

High Pass Filter		
-1	-1	-1
-1	12	-1
-1	-1	-1

Table 3 High pass filter 3x3

4.5.1.3 Laplacian Filter

The Laplacian filter is another image detail sharpening filter that works well for noise-free images. This filter subtracts the brightness values of the four neighboring pixel from the central pixel. The result of applying this filter is to reduce the gray level to zero (Table 4).

Laplacian Filter		
1	-2	1
-2	5	-2
1	-2	1

Table 4 Laplacian filter 3x3

4.5.2 Non-Linear Filter

4.5.2.1 Median Filter

The median filter is used to remove impulse noise spikes from an image and thus smoothing the image. Impulse noise spikes appear as bright or dark pixels randomly distributed throughout the image. Noise spikes are normally significantly brighter or

darker than their neighboring pixels and can easily be found by comparing the median value of a group of input pixels [4]. The Median filter is the most common rank filter, in which we select the middle-ranked value from a neighborhood as our output value.

4.6 Edge Detection

The detection of object boundaries is one of the most fundamental problems in image understanding. Edge Detection is useful for locating the boundaries of objects within an image. Any sudden change in image frequency over a relatively small area within an image is defined as an edge. Location and recognition are far from trivial, because noise and other uninteresting image features can also generate edges [4] [**26**] [**27**].

4.6.1 Sobel Edge

The Sobel operation extracts all of the edges in an image, regardless of the direction. With the Sobel edge is necessary to take two convolutions for each pixel. Typically, this is for a vertical edge and horizontal edge.

We then take the square root of the sum of the squares for each of the convolution results and use this for the output image. It is typical to use a smoothing pre filter before the Sobel edge detection and to use a thresholding afterward. The Sobel edge requires two kernels for the horizontal and vertical edges shown in Table 5 .

Vertical Mask		
-1	-2	-1
0	0	0
1	2	1

Horizontal Mask		
1	0	-1
2	0	-2
1	0	-1

Table 5 Kernels to Sobel Edge

4.6.2 Roberts Edge

The Roberts edge operation extracts edges in an image by taking the combined differences of directions at right angles to each to each other to determine the gradient. The resulting image appears as a fairly-coarse directional outline of the objects within the image. This operation uses the two masks shown in Table 6.

Vertical Mask		
-1	0	0
0	1	0
0	0	0

Horizontal Mask		
0	0	-1
0	1	0
0	0	0

Table 6 Kernels to Roberts Edge

4.6.3 Prewitt Edge

The Prewitt edge operation extracts the north, northeast, east, southeast, south, southwest, west, or northwest edges in an image. The resulting image appears as a directional outline of the objects within the image. This operation uses the two masks shown in Table 7.

Vertical Mask		
-1	-1	-1
0	0	0
1	1	1

Horizontal Mask		
1	0	-1
1	0	-1
1	0	-1

Table 7 Kernels to Prewitt Edge

4.6.4 Frei and Chen Edge

The Frei and Chen edge operation, when compared to the other edge enhancement, operations, is more sensitive to a configuration of relative pixel values independent of the brightness magnitude. This operation uses the two masks shown in Table 8.

Vertical Mask		
-1	-1.414	-1
0	0	0
1	1.414	1

Horizontal Mask		
1	0	-1
1.414	0	-1.414
1	0	-1

Table 8 Kernels to Frei and Chen Edge

Chapter 5

Overview of the JSIM Prototype System

This chapter presents a general description about JSIM Prototype, we also describe about the architecture, history list operator, encapsulation, custom operator, and Metadata. Finally we realize a comparison between JSIM and ImageJ.

5.1 JSIM Prototype

JSIM is an acronym for Java-based SAR image analysis tool environment. JSIM was designed and developed using the Java programming language. Considering advances in information technologies, we envisioned to offer an open-source application, platform independent tool environment, for analyzing and manipulating SAR images. The resulting application allows access to local and remote image-data in a variety of file formats [28].

The tool-environment presented here provides friendly user interfaces, where it is possible to search, manipulate and view images using different image analysis operators such as *filtering, detection edges, enhancement, convolution, correlation, FFT* (Fourier Fast Transform), as well as basic operators such as *load, save, zoom in, zoom out, scale,*

rotate and flip. It allows also multiple files to be opened simultaneously for both viewing and editing.

We are also providing this java-based image analysis tool environment a degree of portability permitted in a given workstation network configuration. The tool runs on various platforms and can be used through the Internet. Figure 7 shows the main graphics user interface of JSIM.

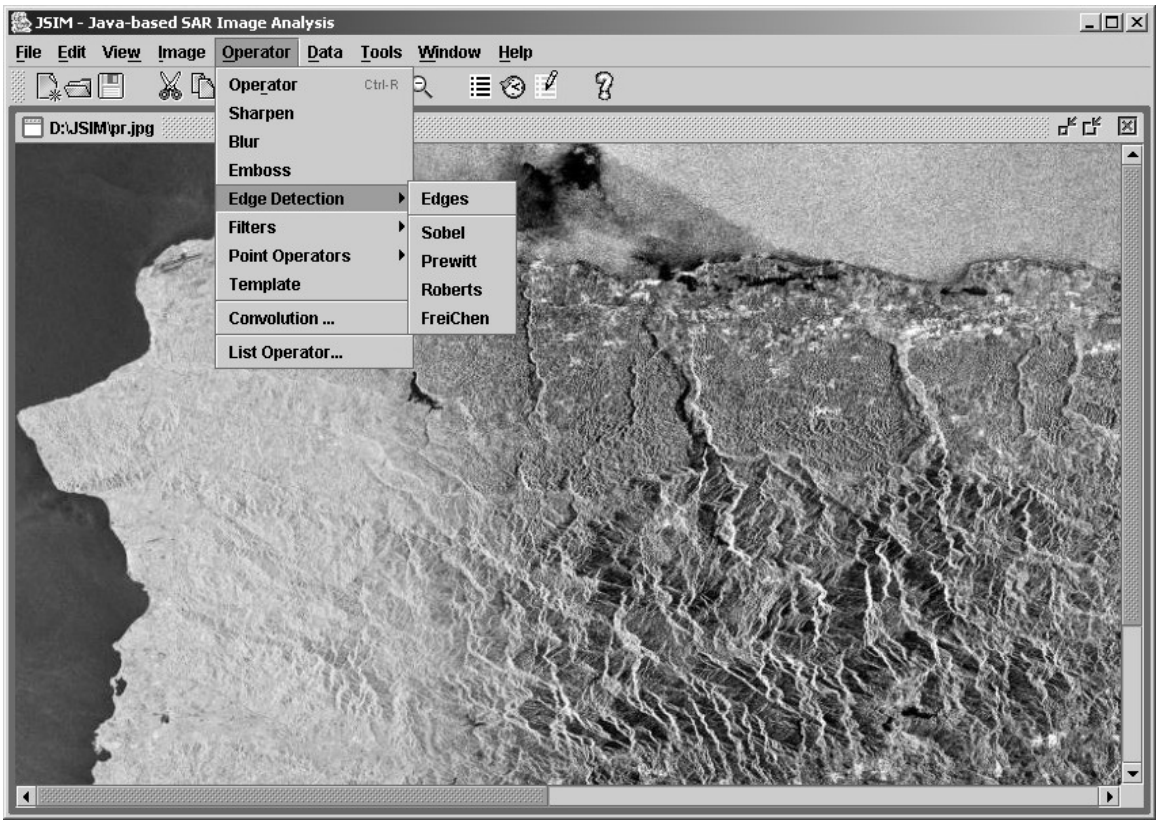


Figure 7 Graphics User Interface of JSIM

5.2 JSIM Network Architecture

One of the important advantages of Java is its network friendliness. The Java core contains many features that help us develop network-based applications. The Remote Method Invocation (RMI) is a distributed-computing architecture. It enables communication between Java applications. With the RMI architecture, an object in one application can invoke a method in another application running on a different Java Virtual Machine.

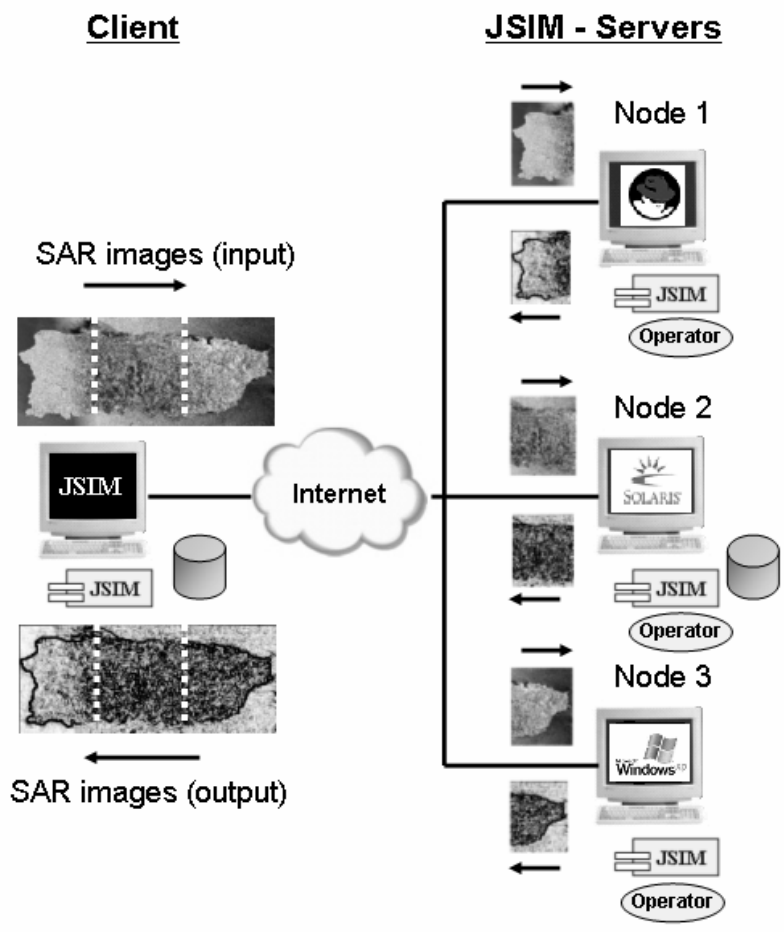


Figure 8 JSIM Architecture

In this work we use the client-server paradigm. Internet-based applications are built in this way, as are many high-end applications [4]. With the client-server approach, multiple clients from a local or a remote machine can access the same application at any time. The clients have the GUI to use the operators provided in this tool. Figure 8 shows the architecture of the application to be used through Internet.

The components of JSIM are:

1. **JSIM - Server**: set of Java-servlet application.
2. **Client Access**: interacts with user application.
3. **Image Database**: contains the SAR images, these can be distributed on the network.
4. **Operators Agents**: algorithms to act on SAR images.

Figure 9, shows internal description of the architecture using java servlet to distribute all tasks between a set of JSIM servers.

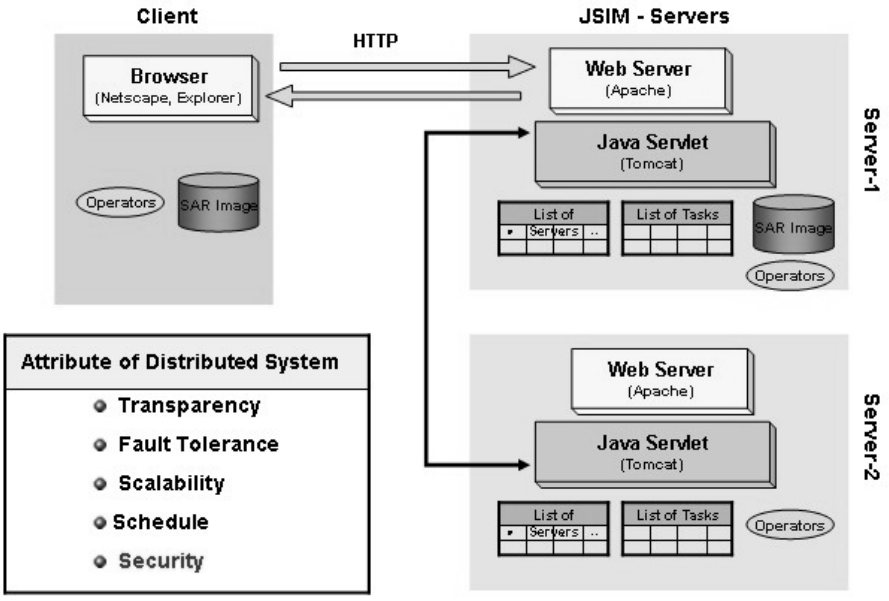


Figure 9 Internal description of the architecture

The architecture of a distributed system offers a wide variety of applications as well as challenges to trying sharing information efficiently over a network of computers.

Figure **10** shows the possible combinations of the elements of a distributed system .The distributed systems are a collection of heterogeneous computers and processors connected via a network to increase the computational power. This collection works closely together to accomplish a common task. A Distributed System consists of a set of processes, executing in different platforms, which communicate via message passing.

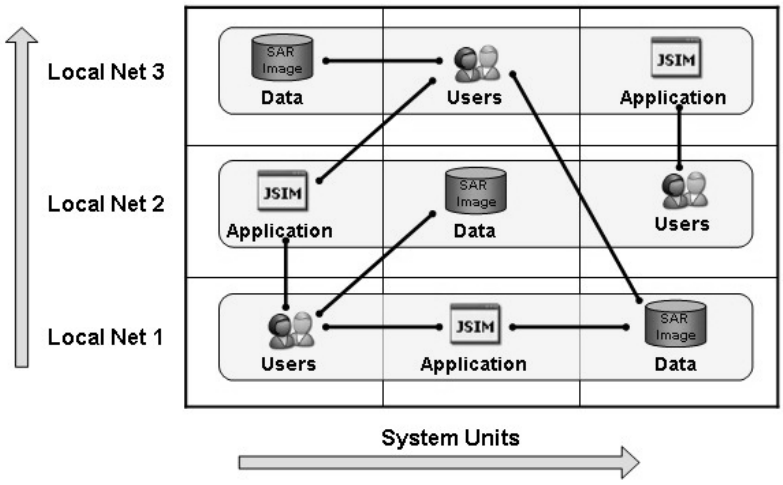


Figure 10 Distributed System Activities

5.3 Operator Agent Environment

The JSIM network architecture presents to user a unique environment with a set of tools for the proper manipulation of object image data in a general setting. We term this environment an *operator agent environment*. Formally, a JSIM operator agent environment is defined as an aggregate of the following components: An object image input data set, an object image output data set, a set of operator agents, a set of composition rules for the operators agents, a set of action rules for the operators agents to

act on the object image input data set in order to produce an object image output data set, and a user interface. Figure 11 depicts an abstraction of the interrelation of the components of an operator agent environment in JSIM network architecture.

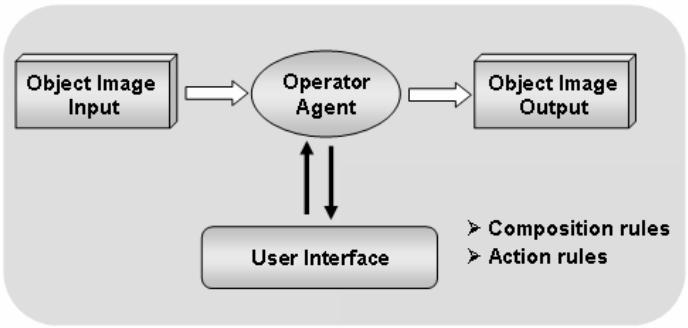


Figure 11 Operator Agent Environments

An environment is usually characterized by its components, the most important of which are:

a. **Object image input data set:** The image data to be processed.

b. **Operator agent:** Functions applied over object image input data set in order to produce some result.

c. **Composition rules:** The way in which operators are combined.

d. **Action rules:** All the actions performed over the operators.

e. **Object image output data set:** The result of applying the action rules.

f. **User interface:** The bridge between the user and the environment tools. The interface makes the environment to be friendly to the final user.

g. **User:** The person or entity making use of the environment tools.

5.4 History List and Operator Encapsulation

The creation of new operators is possible from existing operators which have been initially defined as default. The new operators are encapsulated as a sequential command. The advantage and value adding of this option is the economy of time, when reusing a set of operators.

The steps that should be taken to create an encapsulated operator, starting from a set of the operators by default, are the following: the user should use the operators of the system according to his necessities or the results that he hopes to obtain when combining a set of operators. At the same time, the system will proceed to create a list of the used operators. Once the prospective results are obtained the user can store the sequence of operators used as a new operator for its later use. Figure 12 depicts the manner of how a set of encapsulated operators are used as a new operator to act on a target image.

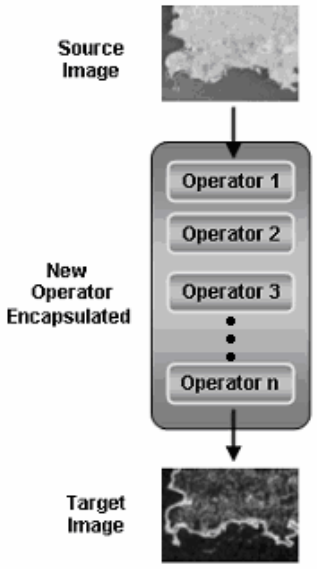


Figure 12 Set of Operators to Encapsulate

From this point on, the set of encapsulated operators are seen by the system as a single entity as presented on Figure 13.

Figure 13 Encapsulated Operators

Figure 14 shows the interface history list. This interface saves the sequence of the operators used with its parameters respectively. This window has four buttons to do the next operations. “Encapsulate Op” button is used to create new operators and encapsulated as a new class operator. “Select all” button is used to select all operators from history list for then encapsulate it. “Clear History” button is used to clear the history list buffer. “Execute” button is used to execute an operator.

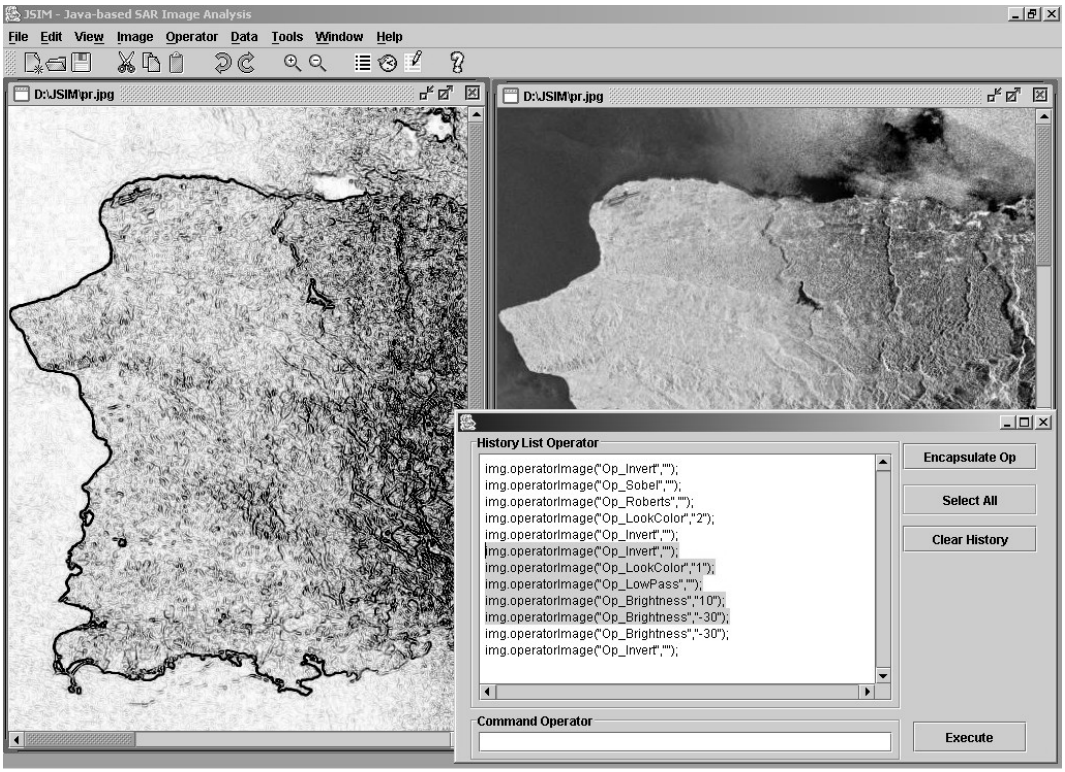


Figure 14 Interface history lists

The next Figure 15 shows the interface list operator. In this interface the user can review all operators of the system. In left side of this window we can navigate or select any operator. Each operator has a brief description and its parameter to use it. Using this interface is possible execute an operator, to do this operation we must select an operator and write the parameters in the text box. Then, click the "Ok" button to execute the operator.

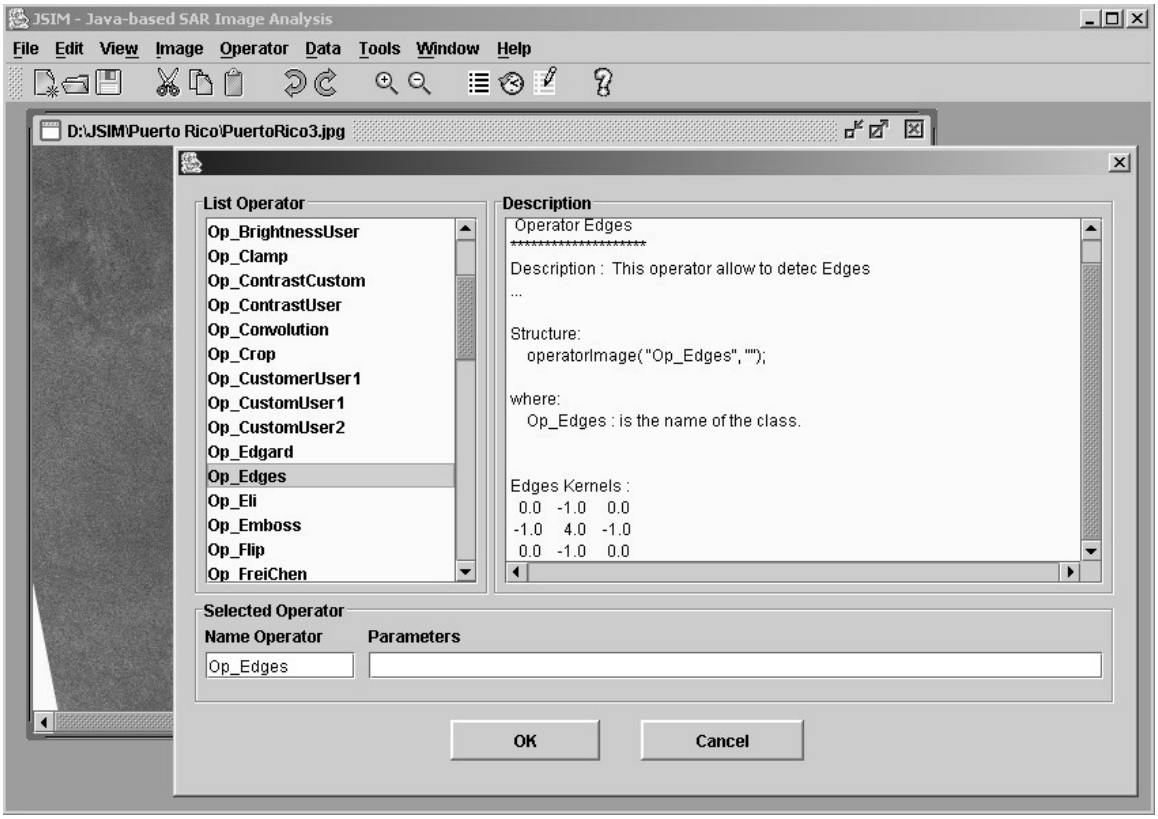


Figure 15 Interface List Operators

5.5 New Operators Customized

The broad field of image processing involves the manipulation and analysis of visual data; fundamental mathematical procedures (known as algorithms) are applied to the data, enabling them to be processed by computers or hardware designed for this purpose.

New algorithms are constantly being developed for procedures such as image encoding, compression, feature extraction, image analysis, and spatial filtering. JSIM allows writing new operator agents (algorithms), implementing new Java classes that are integrated to the existing tool. End-users can add their own customize operations to be used with the SAR images.

One of the many problems that object-oriented software development is meant to solve is the difficulty found when creating dynamic applications that change their behavior at runtime. We use design patterns to solve this problem. The factory method pattern is a model that allows us to dynamically add classes without modifying the program. This pattern helps address this problem of "dynamic creation" by providing a design solution that allows the application to create different objects based on the current system state. Figure 16, shows the design of the pattern to add new operators.

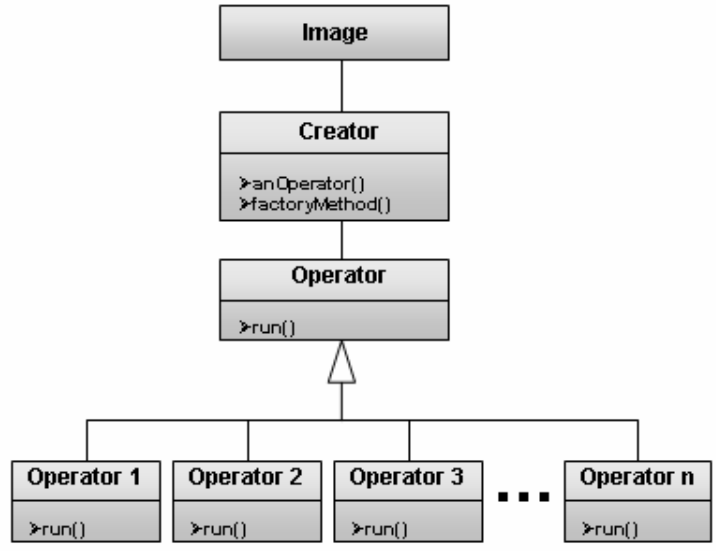


Figure 16 Methodology Pattern for Operator Addition.

5.6 Template class operator design

For the unique option or feature of adding new operators to the tool-environment, a template class was designed as shown in Figure 17. Using this template class the users can create their own algorithms to analyze SAR images. The new operators can be added to the system as part of the library of operators. When the users want to use their operators the system will invoke them dynamically once they have been compiled and attached to the system. The system provides for the interchange of operators among users.

```
public class OperatorTemplate implements Operator {
   RenderedImage image;

   public  String description ()
   {
         String  descrip = " Description about the operator …";
         return descrip;
   }
   public RenderedImage run ( RenderedImage image, parameters String)
   {
         this.image = image;
         int w = image.getWidth();
         int h = image.getHeight();

         // add your own algorithm here

         for (int j = 0; j < h;  j++)
             for (int i = 0; i < w;  j++) {
                   value = input.getSample(i, j, 0);
                   output.setSample(i,j,0,value);
             }
         ...
         return image;
   }
}
```

Figure 17 Template class to create news operators

This template class (see Figure 17) receives an image that has been loaded by the program and transformed into a matrix. Users then should proceed to add their developed algorithms. In this manner, it is simpler for a user to add new algorithms to the system without worrying for additional details.

Figure 18 shows the interface to create new operators for the application. With this interface is possible edit the source code of the operators. In this window we can add or modify the source code, and then we must save the change and compile the operator. This interface shows the errors when we compile an operators. Also, it shows if an operator was created successfully.

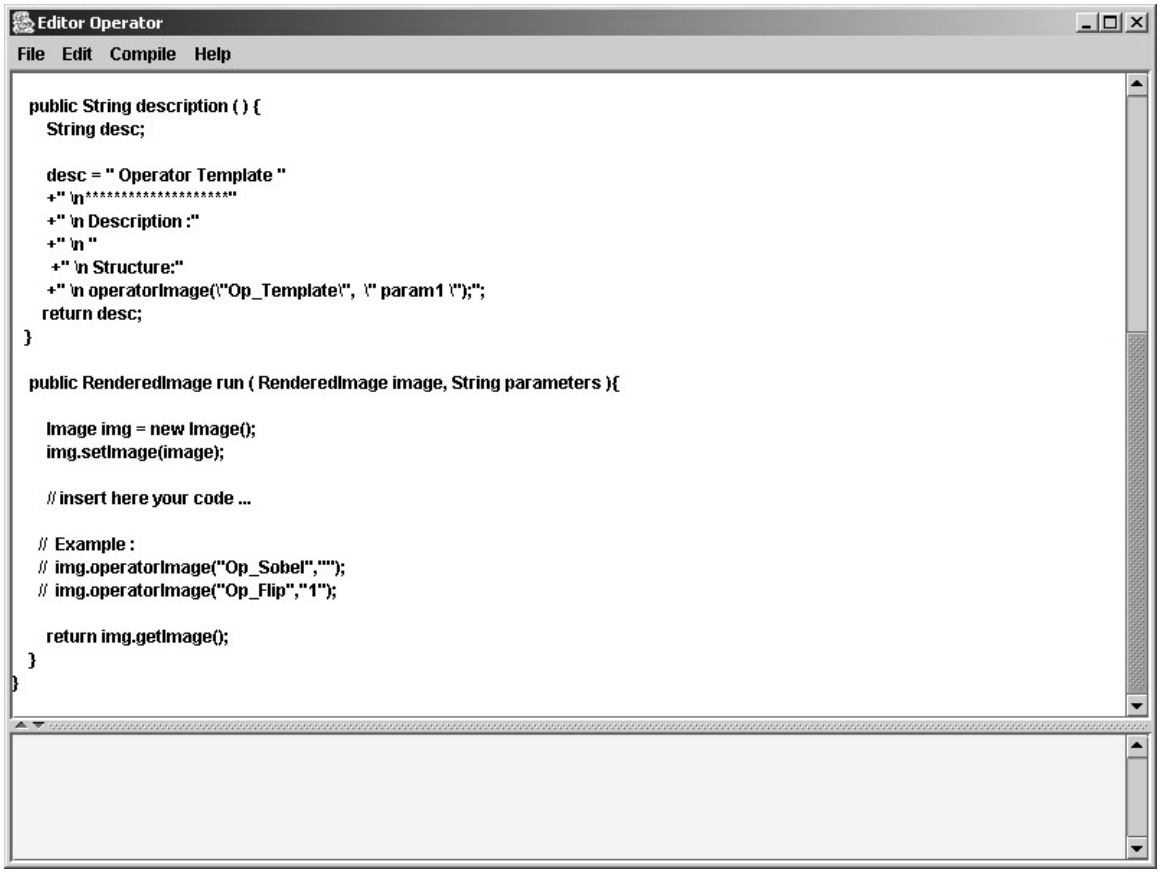


Figure 18 Interface to create new operators

5.7 JSIM Servlet Model

The use JSIM through the Internet will be through the technology of the “servlets” provided by Java. A servlet is a Java program that runs on the server side and is considered an extension of the Web server. The advantage of using servlets is that the client Web browser does not need to support the version of Java or the extensions that a servlet requires.

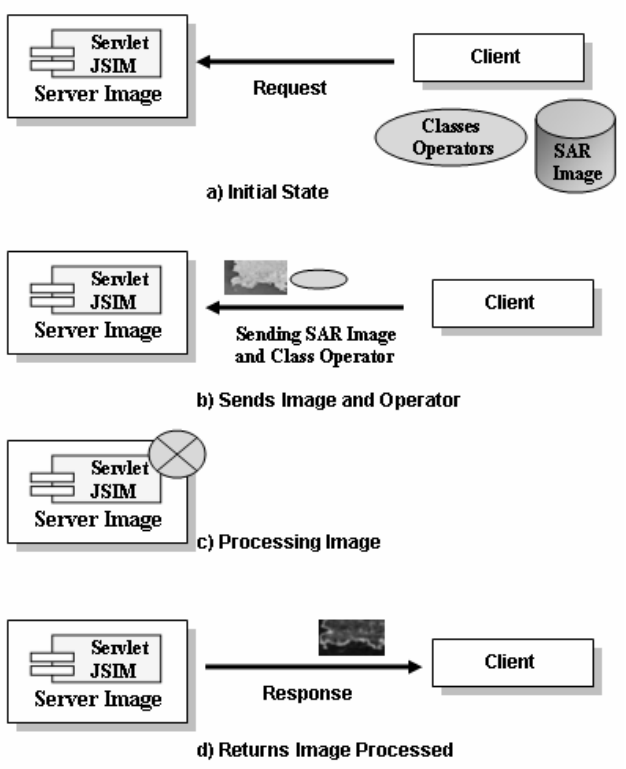


Figure 19 JSIM Using Servlets

The model that we propose to use for JSIM encompasses a server which has the application with a set of basic operators to analyze an image, with an option to use operators provided by the users, and the clients that have SAR images and their own operators. The process to use JSIM is structured in four fundamental steps. The first step

is the initial state where the server is awaiting orders to process and the client has SAR images that a user wants to analyze. In the second step, the client makes a request to process a SAR image and the client sends the image and the operator (optional). In the third step, the program processes the request in the server. Once finished, in the last step, the result is sent to the client. Figure 19 shows a scenario of how the application works using servlets.

5.8 Metadata

Metadata can be defined literally as "data about data," but the term is normally understood to mean structured data about digital and non-digital resources that can be used to help support a wide range of operations. These might include, for example, resource description and discovery, the information about data management and preservation and as well as continuing access.

We used the metadata concept since it help to associate complementary information to SAR images. This information helps to have full advanced knowledge of data characteristics and attributes. Also, the metadata help the end user to organize, find, identify, select, obtain and interpret a SAR image. The metadata is stored as records in a database. To share this information we will use XML due to the flexibility that it offers and its platform-independent syntax. Each SAR image has its own metadata and can include information such as the location of the image, date, resolution, latitude, longitude, format, source, coverage, description, type and other information related. This option is enabled in local mode only. The end users could manipulate metadata if they so desire.

5.9 Comparison between JSIM and ImageJ

The next Table 9 shows the features of JSIM and same time we made a comparison with ImageJ.

Table 9 Features of JSIM and ImageJ

Description	ImageJ	JSIM
Runs multiple platforms		
Linux, Solaris, Mac, Windows	Yes	Yes
Data Types		
8 bit grayscale	Yes	Yes
16 bit unsigned integer	Yes	No
32 bit RBG	Yes	No
File Formats		
TIFF, GIF, JPEG, BMP, ASCII	Yes	Yes
DICOM, FITS	Yes	No
Add new components		
Pluggings	Yes	No
Operators	No	Yes
Image Enhancement		
Smoothing, Edge detection, filtering	Yes	Yes
Regions of Interest		
Rectangular	Yes	Yes
Elliptical, irregular regions, transfer an ROI	Yes	No
Geometric Operations		
Crop, scale, resize, rotate	Yes	Yes
Flip vertically/horizontally	Yes	Yes
Analysis		
Measure Area, Mean, Min and Max	Yes	Yes
Histograms	Yes	Yes
Measurements units	Yes	No
Mode of operation		
Local, Internet	Yes	Yes
Distributed task		
over different computers	No	Yes
Image Display		
Zoom, scrolling	Yes	Yes
Editing		
Cut, copy, undo, paste	Yes	Yes
And, Or, Xor	Yes	Yes
Add text, arrows, rectangles, ellipses, polygons	Yes	No
Metadata		
Local Data	No	Yes
Remote Data	No	No

Chapter 6

Usability Interface Study

6.1 Introduction

The main purpose of the proposed usability test is that the user just focuses his/her attention on the analysis of the images through this application. The objective is that the user does not spend time on training or trying to figure out how to perform a given task. All users' attention must be on creating new algorithms without any difficulty caused by the application interface design. This test is intended to reveal all vulnerabilities of the graphical interface. Once all vulnerabilities are identified, the application will be redesigned to fulfill all deficiencies found.

6.2. Objectives

- Determine the capability of JSIM prototype to manipulate and analyze SAR images.
- Establish if the users understand the functionality of the interface.
- Determine friendliness of the environment for the creation the new operators.
- Evaluate user satisfaction with respect to overall the system.

6.2 User Profile

The level of user expertise and skill will affect an individual's performance. Also, user familiarity with image applications gives an unfair advantage. However, we tried to offset this advantage by training the users about JSIM. In this usability interface study, 20 users of similar skill level participated in the experiments. All users were volunteers their time and were unpaid. All 20 users were graduate students from the University of Puerto Rico – Mayagüez campus.

6.3 Test Methodology

In this experiment the methodology used for each user is outlined in Figure 20.

Figure 20 Sections of the experiment

6.3.1 Introduction.

In this section of the experiment we introduce the user to have a general idea about our research. To accomplish this objective we realized a small presentation to each participant.

6.3.2 Training

In this section, to each participant we realized a demonstration using JSIM tools, in this training we explain about the features of the program and how to use the interfaces. To reduce frustration due to time constraints, we also told them that we did not expect them to complete all tasks. The training time took between 10 a 12 minutes for each user. The user did not perform any practice task. They could ask for help during the experiment, but not ask for assistance in completing a task.

6.3.3 List of Tasks

The abstract tasks used in this usability interface study were:

- Enter to JSIM application
- Use Operators predefined by the program
- Use List Operator Interface
- Use History List Operator Interface
- Create a new Operator with operators predefined
- Use Operator Convolution Customized
- Implement a new algorithm for brightness and contrast in a grayscale image.

6.3.4 Questionnaire

The questionnaire is designed to evaluate the usability of the interfaces through user feedback. The questionnaire is presented to each participant after all tasks have been completed (Appendix C).

Overall reactions: 4 questions to evaluate overall user satisfaction.

Learning: 4 questions to evaluate the learning.

Interfaces: 4 questions to evaluate interface quality.

Terminology and system information: 2 questions to evaluate messages of the system.

System capabilities: 3 questions to evaluate the reliability.

Tasks: 2 questions to evaluate each task.

Miscellaneous: Suggestions or comments.

6.3.5 Interview

An informal interview is held at the close of each experiment. The purpose here is to determine what difficulties the users encountered in using each interface and to extract more about their opinions of usability and recommendations to perform the tool.

6.4 Discussion and Results

This section presents the results obtained of the usability interface study.

6.4.1 Data Analysis

In Table 10, we can observe the time that each user took to complete each task. The task 1 was load the application, then, the task 2, task 3, and task 4 were tasks to interact with the application.

The task 5 and task 10 are similar tasks and their average times are 208.3 seconds and 196 seconds respectively. Therefore, the second time that they made the same task they needed less time in most of the participants. Also, the task 6 and task 8 are similar tasks.

In this case, also they needed less time to complete the task. Finally, task 7 and task 9 are similar tasks, where task 9 was completed more quickly by all participants.

Table 10 Run time of each task by the participants

Users	Time per Task (sec)										Total Time(sec)
	1	2	3	4	5	6	7	8	9	10	
User 1	7	97	94	132	200	45	301	34	231	170	1311
User 2	8	197	106	144	296	80	351	45	221	216	1664
User 3	5	91	68	146	176	43	243	36	134	205	1147
User 4	5	125	67	123	154	48	255	25	88	168	1058
User 5	4	116	75	120	215	45	340	24	185	175	1299
User 6	3	109	72	155	171	55	261	49	162	215	1252
User 7	3	100	75	147	223	40	293	36	191	228	1336
User 8	3	105	60	165	208	38	241	45	149	218	1232
User 9	4	86	57	127	192	49	283	35	170	220	1223
User 10	3	60	73	110	124	42	210	26	165	135	948
User 11	3	100	85	118	158	41	184	31	147	151	1018
User 12	4	104	101	203	298	87	374	47	195	229	1642
User 13	3	86	79	179	244	60	275	80	168	237	1411
User 14	3	88	73	127	168	46	154	38	128	148	973
User 15	5	119	52	177	231	67	289	50	115	218	1323
User 16	3	124	89	148	253	62	349	64	209	224	1525
User 17	5	117	125	150	190	51	257	51	106	164	1216
User 18	3	117	91	163	235	88	292	51	210	220	1470
User 19	3	97	72	133	221	65	295	42	128	183	1239
User 20	3	107	80	145	208	55	276	43	163	196	1276
Descriptive Statistic											
Median	3	105	75	146	208	50	280	43	164	210	1264
St. Desv	1.5	26.3	17.6	23.4	44.7	15.3	55.0	13.4	39.5	31.4	198.7
Average	4.0	107.3	79.7	145.6	208.3	55.4	276.2	42.6	163.3	196.0	1278.2
Max	8	197	125	203	298	88	374	80	231	237	1664
Min	3	60	52	110	124	38	154	24	88	135	948

At this time, in Table 10, we present the total run time required for each participant to complete all tasks. The average time was 1278.2 seconds (21 min approximately). The

minimum time was 948 seconds (16 min approximately), and the maximum time was 1664 seconds (28 min approximately).

In the next Figure 21, we can observe the total run time behavior by each participant.

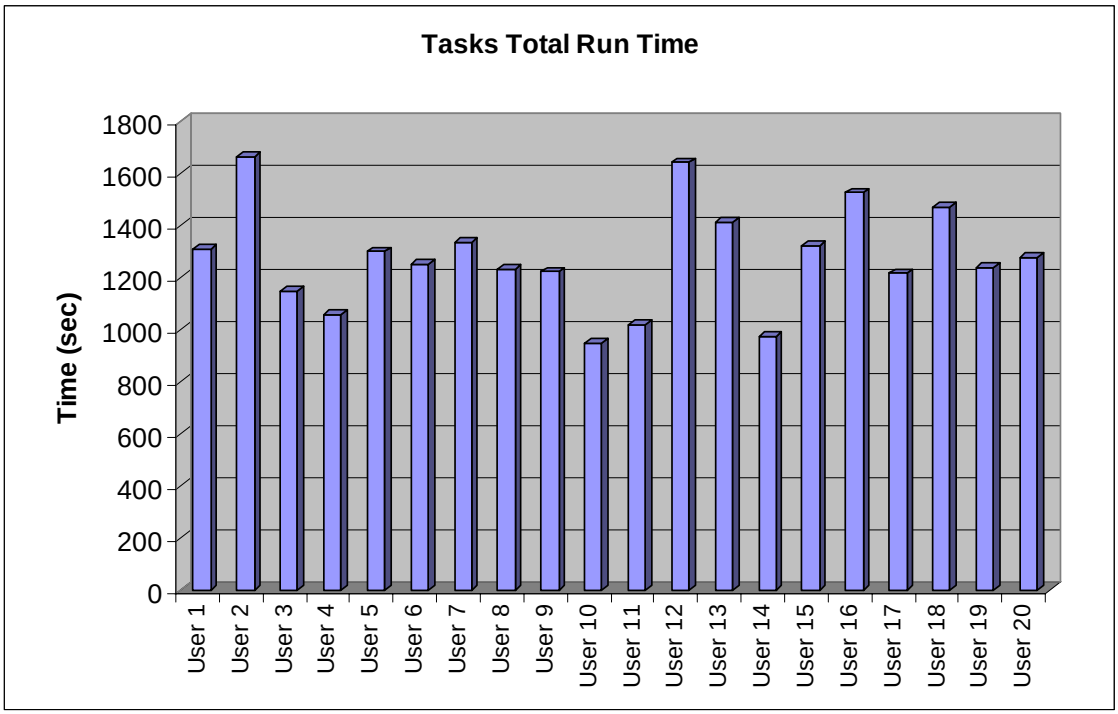


Figure 21 Total run times to complete all task by each participants

6.4.2 Result Evaluation

In this section we analyze the results obtained of the questionnaire that was given to each participant after all task have been completed. In the next Table 11 we present a summary of the results from 16 assertions formulated to the participants. The questionnaire has six parts to know the opinion about the system. The scale was from 1 to 5, where 5 is the most positive value to evaluate each assertion, 3 is a neutral position respect to each assertion, and 1 is the most opposite value to respond the assertions.

Table 11 Questionnaire results by each participant

User	Affirmations															
	Overall Reactions				Learning				Interfaces			Terminology		System Capabilities		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
User 1	4	4	5	5	5	5	5	4	5	5	5	5	5	4	5	5
User 2	4	5	4	5	4	5	4	5	4	4	5	5	4	4	5	5
User 3	5	4	4	5	5	4	5	5	4	4	4	5	5	5	5	3
User 4	4	5	5	5	4	5	4	4	3	5	4	5	5	4	5	5
User 5	4	5	5	5	5	4	4	4	4	3	4	4	3	3	4	4
User 6	4	5	5	4	4	4	4	4	4	4	5	5	5	3	5	3
User 7	5	4	4	4	5	5	4	4	5	5	5	4	4	4	5	5
User 8	3	4	4	4	4	4	4	5	4	4	3	4	4	5	5	4
User 9	4	5	5	4	4	5	5	4	4	5	4	5	4	3	5	4
User 10	4	4	5	4	4	4	4	4	4	4	4	5	5	3	5	3
User 11	5	5	5	5	5	5	5	5	5	5	5	5	4	5	5	5
User 12	4	4	5	4	4	5	4	4	3	4	4	4	3	3	4	4
User 13	4	4	4	4	4	4	4	4	4	4	4	4	4	2	4	5
User 14	5	5	5	5	5	5	4	4	4	4	5	5	5	4	4	2
User 15	5	5	4	4	5	5	4	5	4	5	4	4	2	3	4	5
User 16	4	5	5	5	4	5	5	5	5	5	5	5	4	3	5	3
User 17	4	3	4	4	3	5	3	4	2	3	3	3	2	1	3	5
User 18	5	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5
User 19	4	4	4	3	4	4	4	3	4	4	4	4	5	3	4	3
User 20	4	4	5	4	4	4	5	4	4	5	5	5	5	4	5	4
Descriptive Statistics																
Median	4	4	5	4	4	5	4	4	4	4	4	5	4	3.5	5	4
St. Desv	0.6	0.6	0.5	0.6	0.6	0.5	0.6	0.6	0.8	0.7	0.7	0.6	1.0	1.1	0.6	1.0
Average	4.3	4.4	4.6	4.4	4.4	4.6	4.3	4.3	4.1	4.4	4.4	4.6	4.2	3.6	4.6	4.1
Max	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Min	3	3	4	3	3	4	3	3	2	3	3	3	2	1	3	2

Now, we present each assertion formulated with their own evaluation and graphs for those assertions. Whit these answers for each assertion we can respond to each objective of this usability interface study.

6.4.2.1 Overall Reactions

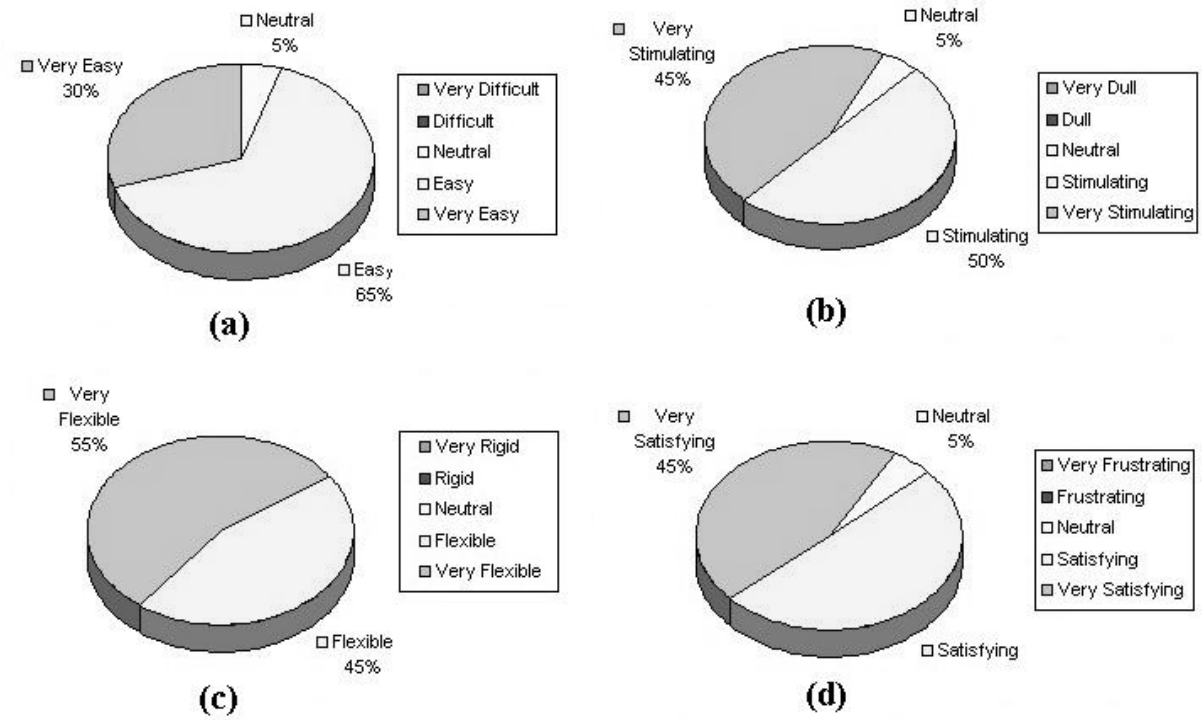


Figure 22 Overall Reactions about JSIM application

Assertion 1: Easy to use.

In Figure 22 (a), we can observe that 30% of the participants say that JSIM is very easy to use. 65% of the participants indicate that is easy to use and only 5% of the participants have a neutral position. In general, the JSIM application is easy to use.

Assertion 2: Stimulating to use.

In Figure 22 (b), we can observe that 45% of the participants were very stimulating, 50% of the participants were stimulating to use the system, and 5% of the participants have a neutral position.

Assertion 3: Flexible to use.

In Figure 22 (c), we can observe that JSIM is very flexible for a 55% of the participants, and flexible for a 45% of the participants.

Assertion 4: Satisfying to use.

In Figure 22 (d), we can observe that 45% of the participants were very satisfying, 50% of them were satisfying. And only 5% of them have neutral position. In general JSIM satisfied their expectative.

6.4.2.2 Learning

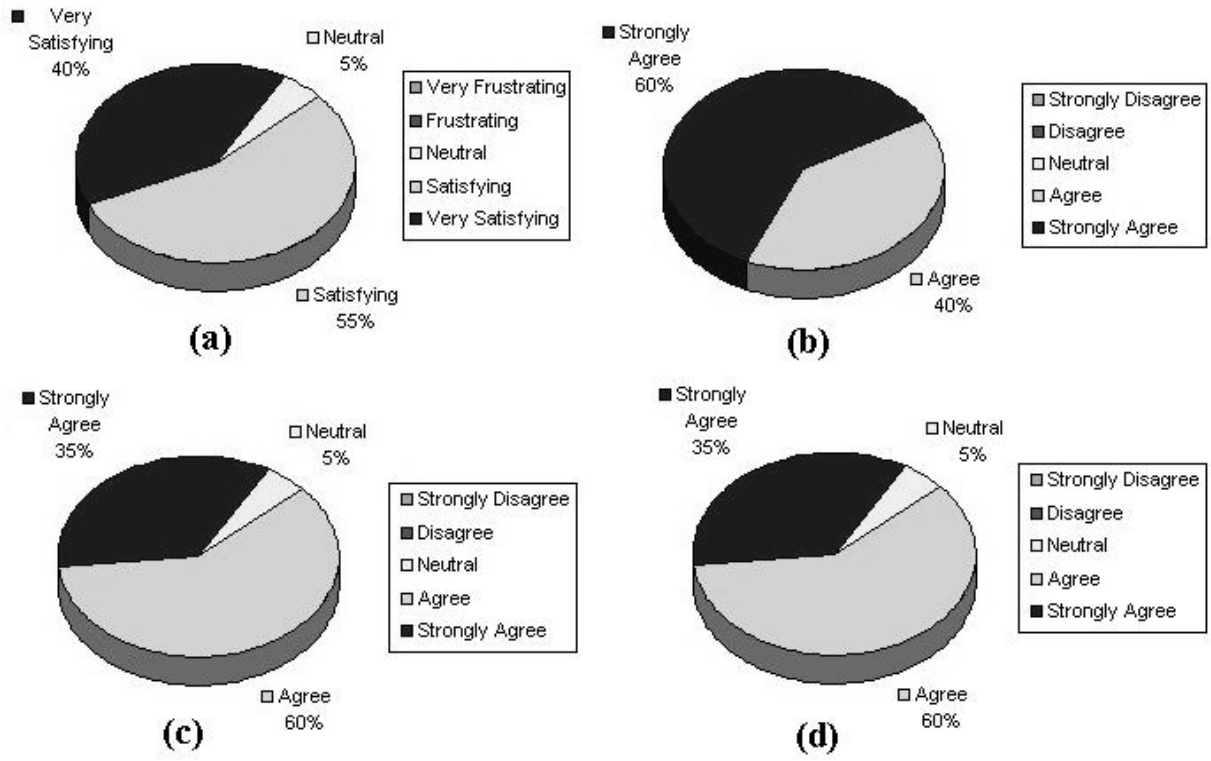


Figure 23 Learning with JSIM application

Assertion 5: Learning to operate the system

In Figure 23 (a), we can observe that 45% of the participants were very satisfying, 55% of them were satisfying, and only 5% of them have a neutral position. Therefore, the majority of the participants do not have problem to operate the system.

Assertion 6: Helping to create new operators

In Figure 23 (b), we can observe all participants affirm that JSIM help them to create new operators.

Assertion 7: Exploring new features

In Figure 23 (c), we can observe that 35% of the participants were strongly agree, because JSIM permit to manipulate new features such as create new operators, view history list and reused the operators in other SAR images.

Assertion 8: Helping to learn more about image processing

- In Figure 23 (d), we can observe that more 95% of the participants were agreeing, because, they learned more about image processing.

In general, we can conclude that JSIM interface is friendly and easy to use for the users, because the answers that we obtained in the section “overall reaction” the participants were satisfied and stimulated to use the application. Also, the majority of them indicated that the system is flexible to work. At this time, the participants affirm that JSIM help them to create new operators for the system. The recommendations were

valuable, because we reveal some vulnerability of the graphical user interface. These observations were corrected and new ideas were implemented to enhancement the prototype. We recommended that is necessary to apply more test with other users to validate our prototype. The sample only was 20 participants; all of them were graduate student of Computer engineering.

Chapter 7

Conclusions and Future Work

In this chapter we present the conclusions and future work about our research. In the section about conclusions, we describe a general conclusion about JSIM prototype and we mention the results obtained in the usability interface study. To conclude, we present the future work, where still there is much to make to improve the prototype and to realize other probe to evaluate the performance of the application.

7.1 Conclusions

In this thesis we have presented a tool environment for image analysis of synthetic aperture radar data. The tool environment was designed and development using Java programming language. We have proposed to use The Java™ Technology to guarantee protocol and platform independence; thus, our basic prototype runs over a heterogeneous environment without inconveniences, at the present time. We believe that this tool will assist scientists and engineers in their SAR image analysis efforts. This prototype provided basic functionality to realize image analysis and it has set operators to start in image processing.

The creation of new operators is possible from existing operators which have been initially defined as default. The new operators are encapsulated as a new class in java. The advantage and value adding of this option is the economy of time, when reusing a set of operators. With this option the users can save as operators the results obtained and apply in another SAR images.

We conducted a usability interface study to validate our prototype. With this study we can determine all vulnerabilities. The recommendations realize by the participants were great utility to perform the prototype.

In summary, our usability interface study demonstrated that JSIM prototype has a friendly user interface, the organization of the main menu help to interact and to use all options of the prototype. Also, the users were stimulated and satisfied with the features provides to create and reuse operators. The users don't have problem in learning to operate the prototype. At this time, the users expressed that using the prototype is possible to learn more about image processing.

Finally, we developed a web site with information and related issues about JSIM application, from this site is possible download the prototype and the source code.

7.2 Future Work

The main limitation of this study was the small sample that we had available to make the usability interface study. Hence, as part of the future work it is necessary to extend the usability test with more users and different backgrounds.

The next step is to extend the prototype and we have some future work to realize in order to improve the quality of the prototype:

- We expect to implement a set of algorithms to analyze other types of images in addition to SAR images. These algorithms will be added as operators to the JSIM network library.
- JSIM will be used to assist in a comprehensive study of land regions in the Greater Antilles. Hence, we must implement operators of high level to aids to land classification.
- We expect to probe the prototype over a cluster of computers to know the performance in this architecture.
- Implement a web site to add, download and interchange new operators developed by other users of the system.
- Create an interface with Matlab to use all functions provides for this products. This option will permit to users reused the algorithm created in this environment.

References

[1] Lawrence H. Rodrigues, “Building Imaging Applications with Java”, Addison-Wesley, USA, 2001.

[2] Yves Voirin, Goze B. Benie, Dong Chen He, Ko Fung, and Kalifa Goita, “A Forest Map Updating Expert System based on the Integration of Low Level Image Analysis and Photointerpretation Techniques”, 2002 IEEE International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, June 2002.

[3] Hellwich O, Heipke C. Wessel B., “Sensor and data fusion contest: information for mapping from airborne SAR and optical imagery”, Geoscience and Remote Sensing Symposium, IGARSS ’01, 2001.

[4] Java Advanced Imaging JAI, SUN Microsystems, http://java.sun.com/products/java-media/jai, USA, November 1999.

[5] Research System Inc., Environment for Visualization Images “ENVI”, http://www.rsinc.com/envi/index.asp,USA, 2000.

[6] Environmental Systems Research Institute ESRI, GIS and mapping software, http://www.esri.com, USA, 2002.

[7] PCI Geomatics, htpp://www.pcigeomatics.com, Canada, 2002.

[8] Khoros Pro 2001 Integrated Development Environment, Khoral Inc. Alburquerque, New Mexico, USA, [Online] (visited2003).

[9] Don Murray, Bill Hibbard, Tom Whittaker, and James Kelly, "Using VisAD to Build Tools for Visualizing and Analyzing Remotely Sensed Data", IEEE 2001 International Geoscience and Remote Sensing Symposium, Sydney, Australia, July 2001.

[10] Seng Chuan, Tay and Kim Hwa, Lim, "Web-Based Processing for Remotely Sensed Data", 2002 IEEE International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, June 2002.

[11] Don Murray, Russ Rew, Steven Emmerson, Jhon Caron, Stuart Wier, and David Fulker, "Unidata's MetApps Project – New ways, in Java, of Exploring Remotely Sensed Data", IEEE 2001 International Geoscience and Remote Sensing Symposium, Sydney, Australia, July 2001.

[12] Stefano Nativi, Paolo Mazzetti, Lorenzo Bigagli and Dino Giuli, "Interoperability Federated System for the Scientific Community working in the EOS sector", IEEE 2001 International Geoscience and Remote Sensing Symposium, Sydney, Australia, July 2001.

[13] G. Walter, F. Warmerdam, and P. Farris, "An Open Source Tool for Geospatial Image Exploitation", 2002 IEEE International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, June 2002.

[14] Do-Hyun Kim, and Min-Soo Kim, “Web GIS Service Component Based On Open Environment”, 2002 IEEE International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, June 2002.

[15] Wayne Rasband, National Institute of Health, Bethesda, Maryland, USA, [Online] (visited 2003).

[16] JVT, Java Vision ToolKit, Mark W. Powell, University of South Florida, Florida, USA,[Online] (visited2003).

[17] Yves Voirin, Goze B. Benie, Dong Chen He, Ko Fung, and Kalifa Goita, “A Forest Map Updating Expert System based on the Integration of Low Level Image Analysis and Photointerpretation Techniques”, 2002 IEEE International Geoscience and Remote Sensing Symposium and the 24th Canadian Symposium on Remote Sensing, Toronto, Canada, June 2002.

[18] Xu Jiang, Zhu Minhui, Wu Yirong, Peng Hailiang, “Parallel programming in SAR image processing”, Geoscience and Remote Sensing Symposium 1999 IGARSS 99, Proceedings I 1999 International, 1999.

[19] D. Paudyal, A. Eiumnoh, J. Aschbacher, “Textural information in SAR images for land-cover applications”, Geoscience and Remote Sensing Symposium, 1995 IGARSS ’95. Quantitative Sensing for Science and Applications, July 1995.

[20] J. Le Caillec, R. Garello, B. Chapron, “Detection of nonlinearity in sea surface SAR imaging process using bispectrum method estimation”, Acoustics, Speech, and Signal Processing, 1995. ICASSP-95, May 1995.

[21] R. Fjortoft, A. Lopes, P. Marthon, E. Cubero-Castan, “Different approaches to multiedge detection in SAR images”, Geosciences and Remote Sensing, 1997. IGARSS ’97. Remote Sensing – A Science Vision for Sustainable Development, 1997.

[22] Li Wang, Deng-Che He, A. Fabbri, “Textural filtering for SAR image processing”, Geosciences and Remote Sensing, IEEE Transactions, 2000.

[23] D.D. Giusto, G. Vernazza, “On the use of virtual information in SAR image analysis”, Multidimensional Signal Processing Workshop, Sep 1989.

[24] Dilia Rueda, Domingo Rodríguez, ”A SAR Algorithm Development Environment Using Matlab”, CRC-99, Computing Research Conference, Mayagüez Puerto Rico, December 1999.

[25] P. Bolon, J. Chanussot, I. Issa, P. Lambert, “Comparison of prefiltering operators for road network extraction in SAR images” Image Processing 1999, ICIP 99. International Conference, 1999.

[26] Efford Nick, “Digital Image Processing”, Pearson Education Limited 2000, British Library Cataloguing in Publication Data, USA, 2000.

[27] Tamal Bose, “Digital Signal and Image Processing”, Library of Congress Cataloging in Publication Data, USA, 2004

[28] Huallparimachi Carlos, Domingo Rodríguez, Manuel Rodríguez, "Design and Development of a Java-based Distributed System Tool for Synthetic Aperture

Radar Image Analysis", International Conference on Computer Science and Technology, CST, IASTED-03, Cancun, Mexico, May 2003.

[29] Huallparimachi Carlos, Domingo Rodríguez, “Java-based Tool for Synthetic Aperture Radar Image Analysis ”, IEEE Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis,ISPA03, Rome, Italy, September 2003.

Appendix A – Instruction Test

INSTRUCTION TEST

TASK 1

Enter to JSIM Application

- Select Icon JSIM to access, or
- Open a DOS Window (write “cmd” and press enter);
  - Go to D:\JSIM
  - Write java JSIM

TASK 2

Use Operators

- Open an Image (PuertoRico1.jpg)
- Select image
- Apply the next operators
  - Crop (Crop Margins Top = 0, Left = 0, Right = 300, Bottom = 300)
  - Rotate (90 grades)
  - Flip Horizontal
  - Invert
  - Brightness

TASK 3

Use List Operator Interface

- Open an Image (PuertoRico2.jpg)
- Select image
- Open Window “List Operator…”
  (Go to main menu to Operator ⟶ List Operator…)
- Select in the list Operator Sharpen
- Read the instruction about the parameters
- Write the parameters in the “text box parameters”
- Apply Operator (press button “Ok”)
- See changes in image output.

TASK 4

Use History List Interface

- Open an Image (PuertoRico3.jpg)
- Select image
- Use the next operators
    - Flip Vertical
    - Crop (Crop Margins Top = 0, Left = 0, Right = 300, Bottom = 300)
    - Brightness
    - Filter Low Pass
    - Edges Detection – Sobel
- Open Window "History List…"
  (Go to main menu to Tools ⟶ History List…)
- Select in the window "Text box – Command Operator"
- Press key "arrow up"(↑) and "arrow down"(↓) to navigate between Operators used.
- Select Operator Sobel (img.operatorImage("Op_Sobel","");)
- Press Button "Execute"
- See changes in image output.

TASK 5

Create a New Operator to Detect Edges (Sobel)

- Select History List Window
- Select Operators used in the Task 4
- Press Button " Encapsulate Op" to create a new java class
- In the "source code" changes the name class to "Op_CustomUser1"
- In the method "description()" add the text "This custom operator is used to detect edges".
- Save the java file as "Op_CustomUser1.java"
- Compile the "source code"
- Open an Image (PuertoRico3.jpg)
- Open Window "List Operator…"
- Select the new Operator ("Op_CustomUser1");
- Apply this operator over image selected.

TASK 6

Use Operator Convolution Customized

- Open an Image (PuertoRico4.jpg)
- Select Convolution Operator
- Use the next Kernel-Convolution

$$KernelConvolution = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

- Write the values in the matrix Kernel
- Apply Operator (press button “OK”)
- See changes in image output

TASK 7

Implement a new Algorithm for brightness of a grayscale image

- Go to main menu “Tools” ⟶ “Editor Operator…”
- Open a class template to add your code (Open file “Op_Template.java”)
- In the “source code” changes the name class to “Op_BrightnessUser”
- Save the java file as “Op_BrightnessUser.java”
- Algorithm

```
For each pixel do
        pixel[i,j] = pixel[i,j] + 50;
```

- Add your code in the method “Run()”
- Compile the “source code”
- Open an Image (PuertoRico5.jpg)
- Open Window “List Operator…”
- Select the new Operator (“Op_BrightnessUser”);
- Apply this operator over image selected.

TASK 8

Use Operator Convolution Customized

- Open an Image (PuertoRico4.jpg)
- Select Convolution Operator
- Use the next Kernel-Convolution

$$KernelConvolution = \begin{bmatrix} 1 & -2 & 0 \\ 1 & 1 & 1 \\ 0 & -2 & 1 \end{bmatrix}$$

- Write the values in the matrix Kernel
- Apply Operator (press button “OK”)
- See changes in image output

TASK 9

Implement a new Algorithm for contrast in a grayscale image

- Go to main menu "Tools" ⟶ "Editor Operator…"
- Open a class template to add your code (Open file "Op_Template.java")
- In the "source code" changes the name class to "Op_ContrastUser"
- Save the java file as "Op_ContrastUser.java"
- Algorithm

```
For each pixel do
        pixel[i,j] = pixel[i,j] * 1.5;
```

- Add your code in the method "Run()"
- Compile the "source code"
- Open an Image (PuertoRico5.jpg)
- Open Window "List Operator…"
- Select the new Operator ("Op_ContrastUser");
- Apply this operator over image selected.

TASK 10

Create a New Operator to Detect Edges (Prewitt)

- Select History List Window
- Select Operators used in the Task 9
- Press Button " Encapsulate Op" to create a new java class
- In the "source code" changes the name class to "Op_CustomUser2"
- In the method "description()" add the text "This custom operator is used to detect edges".
- Save the java file as "Op_CustomUser2.java"
- Compile the "source code"
- Open an Image (PuertoRico5.jpg)
- Open Window "List Operator…"
- Select the new Operator ("Op_CustomUser2");
- Apply this operator over image selected.

Appendix B – Questionnaire Background Information

Questionnaire

PARTICIPANT BACKGROUND INFORMATION

Participant No._____

EDUCATIONAL EXPERIENCE (Highest level completed):

_____ 2-Year College _____ 4-Year College _____ Masters _____ Advanced

COMPUTER EXPERIENCE

Regularly used a computer?

__ Not at all __ Few weeks __ 2 - 6 mos. __ 6 – 24 mos. __ > 2 yrs

Regularly used an image application (e.g. Photo Editor, Corel ,…)?

__ Not at all __ Few weeks __ 2 - 6 mos. __ 6 - 24mos. __ > 2 yrs

Regularly used Matlab to analyze an image?

__ Not at all __ Few weeks __ 2 - 6 mos. __ 6 - 24 mos __ > 2 yrs

Approximately how many hours per week do you use:

A computer (at work or otherwise)? _____hrs

An image application (e.g. Photo Editor, Corel ,…)?_____hrs

A mathematical application (e.g., Matlab)_____hrs

How would you rate yourself in a scale from 1 (amateur) to 5 (expert) using an image application?

Amateur	1	2	3	4	5	Expert

Appendix C – Final Questionnaire User Satisfaction

Final Questionnaire

Participant No.____

User Satisfaction

A. Overall Reactions to the JSIM

1. Easy to use:

Very Difficult	Difficult	Neutral	Easy	Very Easy
1	**2**	**3**	**4**	**5**

2. Stimulating to use:

Very Dull	Dull	Neutral	Stimulating	Very Stimulating
1	**2**	**3**	**4**	**5**

3. Flexible to use:

Very Rigid	Rigid	Neutral	Flexible	Very Flexible
1	**2**	**3**	**4**	**5**

4. Satisfying to use:

Very Frustrating	Frustrating	Neutral	Satisfying	Very Satisfying
1	**2**	**3**	**4**	**5**

B. Learning

5. Learning to operate the system

Very Difficult	Difficult	Neutral	Easy	Very Easy
1	**2**	**3**	**4**	**5**

6. Helping to create new operators in fewer steps.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	**2**	**3**	**4**	**5**

7. Exploring new features

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	**2**	**3**	**4**	**5**

8. Helping to learn more about image processing

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	**2**	**3**	**4**	**5**

C. Interfaces

9. Organization of the main menu

Very Confusing	Confusing	Neutral	Clear	Very Clear
1	**2**	**3**	**4**	**5**

10. Interfaces simplify the tasks

Very Unhelpful	Unhelpful	Neutral	Helpful	Very Helpful
1	**2**	**3**	**4**	**5**

11. Design of the Interface confusing clear

Very Confusing	Confusing	Neutral	Clear	Very Clear
1	**2**	**3**	**4**	**5**

D. Terminology and System information

12. Use of term throughout system

Very Inconsistent	Inconsistent	Neutral	Consistent	Very Consistent
1	**2**	**3**	**4**	**5**

13. JSIM keeps you informed about what it is doing

Never		Neutral		Always
1	**2**	**3**	**4**	**5**

E. System Capabilities

14. System Speed

Too Slow	Slow	Neutral	Fast	Fast enough
1	**2**	**3**	**4**	**5**

15. System reliability

Very Unreliable	Unreliable	Neutral	Reliable	Very Reliable
1	**2**	**3**	**4**	**5**

16. Experienced and inexperienced user' needs are taken into consideration

Never		Neutral		Always
1	**2**	**3**	**4**	**5**

F. Tasks

17. The description of the task was correct.

Description		**Very Confusing**	**Confusing**	**Neutral**	**Clear**	**Very Clear**
1	Enter to JSIM application	**1**	**2**	**3**	**4**	**5**
2	Use Operators	**1**	**2**	**3**	**4**	**5**
3	Use List Operator Interface	**1**	**2**	**3**	**4**	**5**
4	Use History List Interface	**1**	**2**	**3**	**4**	**5**
5	Create a New Operator	**1**	**2**	**3**	**4**	**5**
6	Use Operator Convolution Customized	**1**	**2**	**3**	**4**	**5**
7	Implement a new Algorithm for brightness of a grayscale image.	**1**	**2**	**3**	**4**	**5**
8	Use Operator Convolution Customized	**1**	**2**	**3**	**4**	**5**
9	Implement a new Algorithm for contrast in a grayscale image	**1**	**2**	**3**	**4**	**5**
10	Create a New Operator to Detect Edges (Prewitt)	**1**	**2**	**3**	**4**	**5**

18. The level of difficult of the tasks.

Description		Very Difficult	Difficult	Neutral	Easy	Very Easy
1	Enter to JSIM application	**1**	**2**	**3**	**4**	**5**
2	Use Operators	**1**	**2**	**3**	**4**	**5**
3	Use List Operator Interface	**1**	**2**	**3**	**4**	**5**
4	Use History List Interface	**1**	**2**	**3**	**4**	**5**
5	Create a New Operator	**1**	**2**	**3**	**4**	**5**
6	Use Operator Convolution Customized	**1**	**2**	**3**	**4**	**5**
7	Implement a new Algorithm for brightness of a grayscale image.	**1**	**2**	**3**	**4**	**5**
8	Use Operator Convolution Customized	**1**	**2**	**3**	**4**	**5**
9	Implement a new Algorithm for contrast in a grayscale image	**1**	**2**	**3**	**4**	**5**
10	Create a New Operator to Detect Edges (Prewitt)	**1**	**2**	**3**	**4**	**5**

G. Miscellaneous

What problems did I have using the interface?

_____

_____

Any other suggestions or comments?

_____

_____

Appendix D – Application Form Data Collection

Application Form - Data Collection
JSIM

User Id : _____
Date : _____
Start Time : _____
End Time : _____

	Description	Number of Intents	Complete		Time
1	Enter to JSIM application				
2	Use Operators				
3	Use List Operator Interface				
4	Use History List Interface				
5	Create a New Operator to Detect Edges (Sobel)				
6	Use Operator Convolution Customized				
7	Implement a new Algorithm for brightness of a grayscale image.				
8	Use Operator Convolution Customized				
9	Implement a new Algorithm for contrast in a grayscale image				
10	Create a New Operator to Detect Edges (Roberts)				

.

Appendix E – JSIM Presentation

Design and Development of a Java-based Distributed System Tool for SAR Image Analysis

Carlos Huallparimachi Suárez, Ms Student
Domingo Rodríguez, PhD., Advisor

March - 2004

Electrical and Computer Engineering Department
University of Puerto Rico, Mayagüez Campus

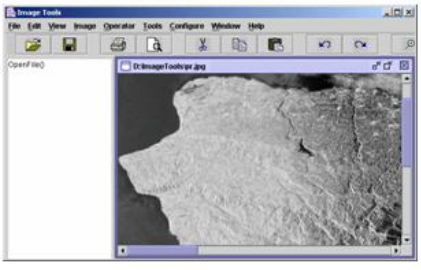
Introduction

Develop a distributed system tool for synthetic aperture radar (SAR) image analysis.

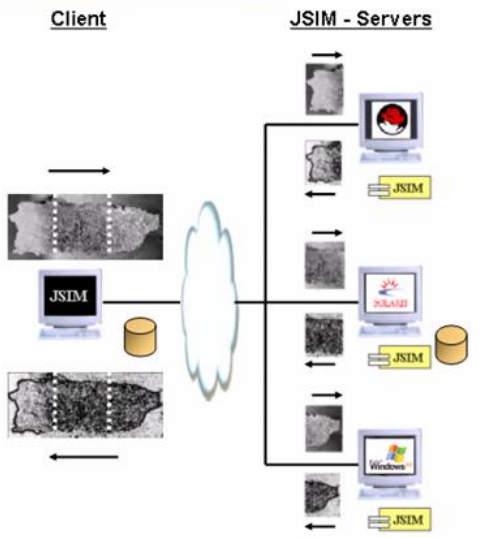
This tool-environment allows to visualize, manipulate, improve, filter, detect edges, reduce the noise on SAR images.

The SAR technology allows obtaining terrestrial images of high spatial resolution, even through cloudy skies.

SAR image data is readily available from many public and private institutions.

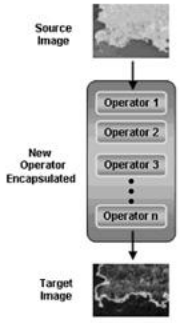


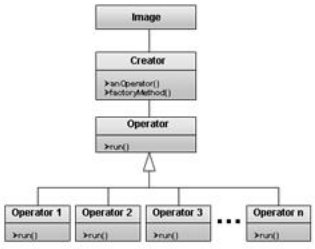
JSIM - Architecture



History List and Operator Encapsulation

- Creation of new operators is possible from existing operators.
- New operators are encapsulated as a sequential command.
- Can reuse a set of operators as a new operator.





- End-users can add their own custom operators.
- A template class was designed to add new algorithms.