DEVELOPMENT OF SUPERCOMPUTER IMAGE PROCESSING SOFTWARE WITH X-WINDOW USER-INTERFACE FOR THE PROCESSING OF THE REMOTELY SENSED DATA

Young-Kyu Yang, Hyun-Ok Nam, Kyoung-Ok Kim, Seong-Ik Cho

Systems Engineering Research Institute
Korea Institute of Science and Technology
1 Eoeun-dong, Yoosung-ku
Taejeon, Korea 305-333

COMMISSION II

ABSTRACT

This paper presents the design concept, the functions, and the examples of utilization of C-ERIMS (Environment and Resources Integrated Management System for CRAY) image processing software package under development on Cray-2 supercomputer system. C-ERIMS is equipped with almost all the commonly used image processing algorithms as well as special features like parallel processing, neural network, 3-D perspective animation, etc. It has interactive true color display capability utilizing x-window on the CRAY supercomputer system under UNICOS operating system.

KEY WORDS: Image Processing, Supercomputer, Neural Network, Parallel Processing, X-window

1. INTRODUCTION

A powerful supercomputer based remote sensing image processing software package is being developed at the Korea Institute of Science and Technology (KIST). KIST have experience of developing image processing software on microcomputer and main frame computers (Yang, et al., 1989a; Yang, et al., 1989b).

This software is code named C-ERIMS which is an acronym for Environment and Resources Integrated Management System for CRAY. C-ERIMS is designed to operate as a complete user-friendly interactive system with integrated software and open Graphic User Interface (GUI), i.e. X Window system.

C-ERIMS system can:

- o directly access satellite images and other remotely sensed data,
- o extract subsections or full scene to disk files,
- o act upon these files to perform the generally available image processing functions, and
- o generate hard copies of processed images.

The philosophy behind the developing C-ERIMS was to take advantage of the computing power and large main memory of supercomputer and provide the resource mangers and other decision makers with a powerful tool necessary for their national scale resource management.

2. BASIC DESIGN CONCEPT

2.1 Software Design

A complete survey for the existing image processing software was performed for maximum efficient design of the C-ERIMS system. The CSADIE (Samayoa, 1983) and GIPSY (Gu and Samayoa, 1988) software packages available on Cray computer system seem to have limited capability due to their batch oriented operation and lack of true color manipulation capability. They were originally developed on smaller computer system such as workstation or Cyber system and later ported to the Cray computer. Other systems including VICAR, ELAS (NASA/ERL, 1988), and LARSYS (Williams, et al., 1976) were also surveyed and analyzed. A comprehensive image processing software was successfully designed to handle full scene satellite imagery in interactive way using X Window protocol.

2.2 X Window System

The X Window system developed at the MIT (Massachusetts Institute of Technology) seems to satisfy the needs of users for high-performance, high functionality, network based system for high-resolution graphics. The purpose of the X Window system is to provide a network-transparent and vendor-independent operating environment for workstation software and to maintain portability with a common programming interface across multiple vendor platforms.

The architecture of the X Window system is based on a client-server model. The X server controls all resources which include windows, bitmaps, fonts, colors, and other data structures used by the window system to enable clients to use and share these data structures transparently. Network transparency implies that applications can run on whatever CPU is most convenient. For example, applications requiring extensive computations can run on a network-connected supercomputer and the result can be displyed on the terminal or workstation.

3. SUPERCOMPUTING ENVIRONMENT

3.1 Cray-2S System

Cray-2s/4-128 supercomputer was used as the platform for software development. It has four CPUs and equipped with 128 mega word SRAM main memory and 40 giga bytes of high speed disk. The clock cycle is 4.1 nano second and its peak performance is 2 GFLOPS (Giga Floating Point Operations per Second). The operating system is UNICOS system which is based on AT&T UNIX System V and enhanced to support both batch and interactive processing in a large scale scientific computer environment. UNICOS supports CAL, the Cray Macro Assembler, and the high level languages including Fortran, C, and Pascal. The Fortran compiler has capability of automatically vectorizing inner DO-loops, and thus provides program optimization functionalty.

3.2 Network

The Cray-2S system is connected to IBM 3083, and MicroVax system through 100 mega bytes/ second HYPER Channel (Figure 1). Local Ethernet network is also connected to Cray system through TCP/IP networking protocol. A remotely located NAS AS/XL V50 and a CDC Cyber 960-31 mainframe computers are connected to Cray system via dual 1.5 mega bits per second T1 communication line. The Unix workstations including Sun, Silicon Graphic's IRIS are connected to Ethernet network and thus provides users with accessibility to any of the existing computer resources including Cray, NAS, Cyber, Vax, and other workstations.

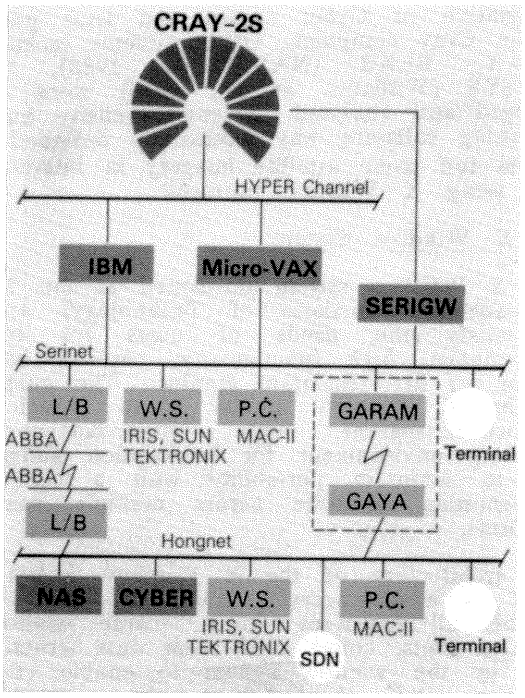

Fig. 1. Supercomputing environment in KIST.

4. IMAGE PROCESSING MODULES

Most of commonly available image procesing functions are being implemented. It includes preprocessing, image enhancement, image transformation, image classification, overlaying and mosaicking, data compression, and some utility functions.

4.1 Preprocessing

The sources of geometric error are instrument error, panoramic distortion, earth rotation and platform instability. The geometric correction process comprises the determination of a relationship between the coordinate system of map and image. It establishes a set of points defining pixel centers in the corrected image that, when considered as a rectangular grid, defines an image with the desired cartographic properties, and the estimation of pixel values to be associated with those points. The cubic, bilinear, and nearest neighborhood method are available as the resampling methods.

4.2 Image Enhancement

Image enhancement is to enhance the weakened image and to increase visible effect. Contrast stretch method (linear, normalized, histogram equalization, exponential, and logarithmic methods) for image enhancement were implemented. Spatial filtering with mean, mode, Roberts gradient, difference operator, and sobel operator were also implemented. FFT (Fast Fourier Transformation), contouring, and color enhancement is also available.

4.3 Image transformation

Various image transformation algorithm is available. They can be summarized as follows:

1) Arithmetic operation:
   - o band ratio
   - o band difference
   - o sum
   - o add

2) Vegetation index
   - o normalized difference
   - o leaf area index
   - o transformed vegetation index
   - o tasselled cap transformation
   - o perpendicular vegetation index

3) Principal Component Analysis

4.4 Classification and Other Algorithms

1) Classification
   - o supervised
   - o unsupervised

2) Terrain analysis
   - o slope-gradient
   - o slope-aspect
   - o shaded relief

3) Three dimensional perspective view (Figure 2)
   - o coordinate transformations
   - o projection transformation
   - o hidden line procedure
   - o scaling

4) Generation of mosaic map (Figure 3)
   - o density level normalization
   - o correction of the density level by overlapped area
   - o seam control point matching

5) Utility routines
   - o data file handling
     - file format conversion
     - band extraction
   - o coordinate conversion
     - UTM versus geographic coordinate
   - o color map manipulation
   - o histogram computation
   - o statistical analysis
   - o image compression

4.5 Time Comparison with Other Systems

Test run of the several algorithms on Cray-2S computer shows more than 50 times improvement in processing speed comparing to the IBM 3083 and VAX system. The algorithms used for test are as follows:

- o GCP correction
- o 3-D perspective view display
- o Maximum likelihood classification

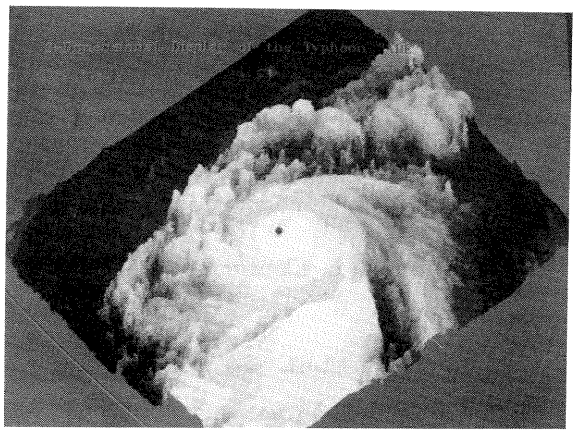

Fig. 2. Three dimensional display of Typhoon "Abe".

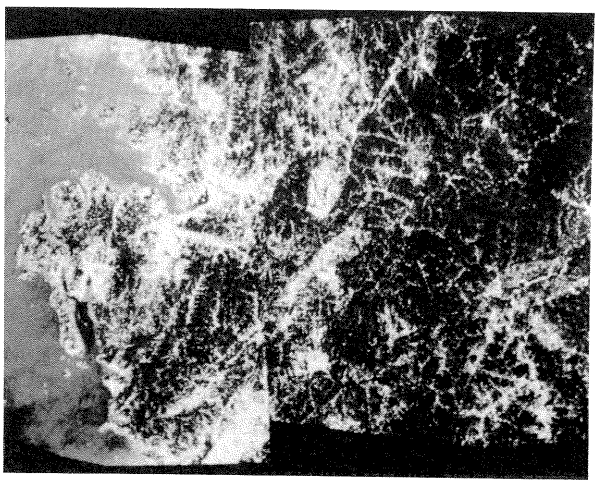

Fig. 3. Mosaic image of middle part of Korea.

5. NEURAL NET ALGORITHM

5.1 Introduction

Recently efforts to adopt neural net algorithm to satellite imagery analysis have been very active (Benediktsson, et al., 1990; Heermann and Khazenie, 1992; Lee, et al., 1990; Ryan, et al., 1991).

A neural net algorithm and software named "atree (Adaptive TREE)" was adopted and implemented in Cray-2S. It was originally developed for PC and workstations by University of Alberta, Canada (Dwelly, 1990). The atree contains the learning algorithm for adaptive logic netwoks, which is the backpropagation algorithm for multilayer feed forward artificial neural networks. The atree algorithm demostrates that for those networks which is consisting of multilayer threshold,its elements can be effectively trained.

It is an important property for logic networks to have ability to generalize their responces to new inputs, presented after training is completed. The successful generalization properties of these logic networks are based on the observation, backed up by a theory, that trees of "two-input" logic gates of types AND, OR, LEFT and RIHGT are very insensitive to changes of their inputs.

5.2 "Atree" Algorithm Concept

An atree (Adaptive TREE) is the fundamental structure used by routines in this software (Dwelly, 1990). This is a binary tree with nodes of two types: (1) adaptive elements, and (2) leaves. Each element takes one of four logical functions, AND, OR, LEFT, or RIGHT depending on initialization or the training it has undergone. The leaf nodes of the tree serve only to collect the inputs to the subtree of elements. Each one takes its input bit from a boolean input vector. The tree produces a single bit as its output. For computing non-boolean outputs, several trees are used in parallel to produce a vector of bits representing the output value.

5.3 "Atree" Implementation

The original routines written in C language by University of Alberta to create, train, evaluate, and print out adaptive logic networks were transported to the Cray computer. The user can create a training set, then create a tree using 'atree_create'. The tree is trained using 'atree_train' and then it can be used to evaluate new inputs using 'atree_eval'. Because a single tree produces only one bit, the programmer must train several trees on the input data, each one responsible for one bit of the output data. This is made slightly simpler by the choice of parameters for 'atree_train' which takes an array of bit vectors as the training set, and an array of bit vectors for the result set. The figure 4 shows its typical description.

6. USER INTERFACE

6.1 Graphic User Interface

X-lib was selected as the graphics tool for C-

ERIMS system. It defines an extensive set of functions and provides complete access and control over the display, windows, and input devices. It seems to become one of the most widely used low-level graphic interface in Unix systems.

Motif was selected as the user interface tool. Many software developers prefer to use one of the higher-level X toolkits since using low-level graphics library to develop application software could be tedious and time consuming. A toolkit should contain a user-interface subroutine library to simplify the design and development of application user interfaces. Motif provides special tool kit feature named widgets which supports human interface graphic objects, such as dialogue box, menu, or mouse-sensitive button.

6.2 Execution of an X program

A minimal X application first sets up a connection to the workstation, then creates a window, creates X resources, maps the window to the screen to make it visible, solicits the input events it is interested in, reads and interprets these input events, and finally generates graphical output.

In order to run an X Window based software, following procedure is necessary.

1) Compilation of an X application program in Cray computer:

```
cc filename -lX11 -lnet
```

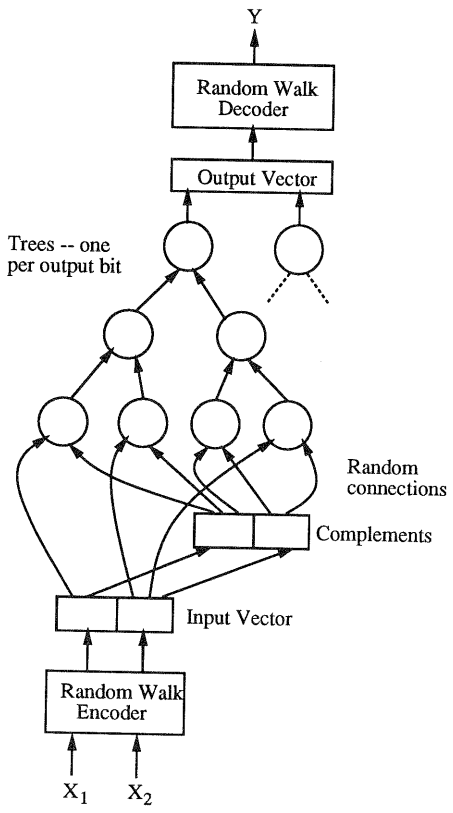

Fig. 4. Description of Atree structure (Dwelly, 1990).

2) Designating the display device:

```
xhost Cray2s
telnet Cray2s
userid: xxx
passwd: yyy
setenv DISPLAY iris000
```

6.3 X Toolkit Programming

In X Toolkit programming, Xt Intrinsics serves as a framework that allows programmers to create user interfaces by combining an extensible set of user interface components. These components are known as widgets, and include menus, dialogue boxes, title bars, scrollbars, selection boxes, labels and push buttons.

X Toolkit application performs several basic steps. These are:

- o Initialize the Intrinsics.
  XtInitialize() establishes a connection to the X server.
- o Create widgets.
  XtCreateWidget(name, class, parent, args, nargs)
- o Register callbacks and event handlers.
  Xt
- o Realize all widgets.
- o Enter the event loop.

The compilation step would be:

```
cc -o filename1 filename2 -lXm -lXt
-X11
```

7. PARALLEL PROCESSING

In order to benefit from the power of vector hardware, existing programs should be rewritten or translated to the parallel form. The existing auto vectorizing compiler does not provide maximum efficient method to discover the parallelism implicit in a Fortran program.

An efficient translation algorithm is being developed as shown in figure 5. The scanner-parser phase converts the input program to an abstract syntax tree that is used as the intermediate form throughout the translation. The pretty printer can reconstruct a source program from the abstract syntax tree; it is used throughout the translator. The vector translation phase consists of three main subphases:

(1) subscript standardization, which encompasses all the transformation that attempt to put subscripts into canonical form;
(2) dependence analysis, which builds the interstatement dependence graph;
(3) parallel code generation, which generates array assignments where possible.

8. CONCLUSION

A Cray-2s based image processing system code named C-ERIMS is being developed. It will

have full scene handling capability taking advantage of high speed and large main memory of the supercomputer. It's software will be composed of the most of the commonly available functions including preprocessing, image enhancement, image transformation, classification, image compression, three dimensional perspective view generation, mosaic map generation and terrain information analysis. Most of the image processing modules are written in Fortran and C. Test run of the some algorithm in C-ERIMS on the Cray showed more than 50 times improvement in processing speed comparing to the IBM and VAX systems.

C-ERIMS has user friendly interactive capability on X-terminal or UNIX workstations connected to Cray system via X Window protocol. The Xlib was adopted as the graphic tool and Motif tool kit provides easy-to-use menu system. It will also be equipped with special features such as neural net algorithm, parallel processing algorithm, and some of knowedge based inference algorithm to enhance the processing capability and the classification accuaracy. C-ERIMS system is expected to be available to the public by the end of 2nd quarter next year (1993).

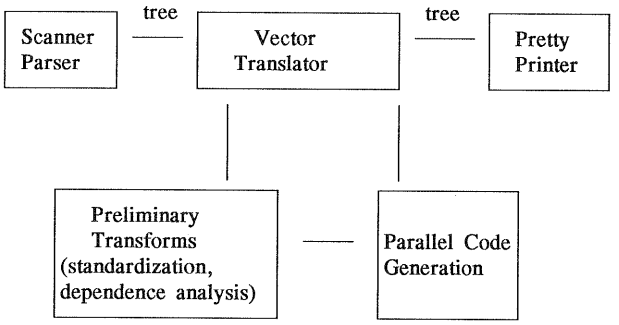

Fig. 5. Overview of Translation Process

9. ACKNOWLEDGEMENTS

The authors would like to thank Cray Research, Inc. for partially supporting C-ERIMS development project through Cray University Grant Program.

REFERENCES

Benediktsson, J. A., P. H. Swain, and O. K. Ersoy, 1990. Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Date, IEEE Transactions on Geoscience and Remote Sensing, 28(4): 540-551.

Heermann, P. D. and N. Khazenie, 1992. Classification of Multispectral Remote Sensing Data Using a Back-Progation Neural Network, IEEE Transactions on Geoscience and Remote Sensing, 30(1):81-88.

Dwelly, Andrew, 1990. An Implementation of Adaptive Logic Networks, University of Alberta, Canada. 25 pg.

Gu, Goson and B. Samayoa, 1988. Conversion of Gipsy Code on Cray Unicos System, The Final Report on 1988 Summer Intern Project in Digital Image Processing, Cray Research Inc., 31 pg.

Lee, J, R. C. Weger, S. K. Sengupta and R. M. Welch, 1990. A Neural Network Approach to Cloud Classification, IEEE Transactions on Geoscience and Remote Sensing, 28(5):846-855.

NASA/ERL, 1988. ELAS: Earth Resources Laboratory Application Software. NASA/ERL NSTL Report 183. NSTL, MS.

Ryan W., P. J. Sementilli, P. Yuen, and B. R. Hunt. 1991. Extraction of Shoreline Features by Neural Nets and Image Processing. Photogrammetric Engineering and Remote Sensing, 57(7):947-955

Samayoa, William, 1983. CSADIE 3.0 for Cray Computers: System Documentation. Cray Research Inc., Mendota Height, Mn. 175 pg.

Williams, G. N., et al., 1976. Conversion of LARSYS III.1 to an IBM 370 Computer. Final Report, Contract NAS9-14514 by Texas A&M Univ. for Earth Observation Division, NASA Johnson Space Center, Houston, TX. 60 pg.

Yang, Y. K., S. E. Cho, et al., 1989a. Development of a Microcomputer Image Processing Systems for Analyzing Satellite and Airborne Sensor Data. MOST National Research Report BS N20622. 173 pg.

Yang, Y. K., S. E. Cho, et al., 1989b. Development of Satellite Image Processing Software on Mainframe Computer. Journal of Korean Society of Remote Sensing, 5(1):29-39.