Image Cover Sheet

CLASSIFICATION	SYSTEM NUMBER
UNCLASSIFIED	518203

TITLE

Extensions to Real-Time Hierarchical Mine Detection Algorithm

System Number:

Patron Number:

Requester:

Notes:

DSIS Use only:

Deliver to: DK

THIS PAGE IS LEFT BLANK

THIS PAGE IS LEFT BLANK

Defence Research and Development Canada
Recherche et developpement pour la défense Canada

DEFENCE R&D DÉFENSE

Extensions to Real-Time Hierarchical Mine Detection Algorithm

Final Report

Sinh Duong and Mabo R. Ito
University of British Columbia

Scientific Authority:
John McFee
Defence R&D Canada – Suffield

Contract Report
CR 2002-121
September 2002

Canada

DEFENCE R&D CANADA - SUFFIELD
RALSTON, ALBERTA

DRES CONTRACT DSS FILE W7702-9-R750/001/EDM

EXTENSIONS TO REAL-TIME HIERARCHICAL MINE DETECTION ALGORITHM

FINAL REPORT

by

Sinh Duong and Mabo R Ito

The University of British Columbia

Vancouver, B.C.

August 2002

Abstract

The Threat Detection Group (TDG) at Defence R&D Canada - Suffield (DRDC Suffield) has undertaken a research program on the feasibility of remote sensing of minefields. One of the projects is the Remote Minefield Detection (RMD) hierarchical algorithm originally developed for single band active airborne infrared imagery (AAII) and later adapted for vehicle mounted passive infrared imagery (FLIR)

The objectives of this contract are: (i) to implement the RMD hierarchical algorithm on a down-scaled version of transputer architecture for real-time mine detection on FLIR imagery, (ii) to develop the high and top levels of the algorithm including expert system and knowledge base, and (iii) to research the possibility to upgrade the current hardware platform to modern advanced computational elements.

Phase 1 and 2 of the contract focused on the implementation of Low and Middle levels of a real-time RMD system on a transputer network to detect mines in real FLIR images. The work first described the hardware requirement for each module of the algorithm, then outlined the software development, and finally presented some preliminary test results.

Although transputers were attractive computing elements 15 years ago when this project started, they have become obsolete. Thus a decision has been made by the Project Authority to implement the RMD hierarchical algorithm on a new hardware platform, namely a network of Intel Pentium-class processors, which was recommended by a study from the University of British Columbia Phase 3 of the contract involved building that PC network, and developing and testing the new software version

Table of Contents

Table of Figures

Glossary

Active Airborne Infrared Imaging (AAII): Active infrared images obtained from monochromatic airborne imagers.

Compact Airborne Spectrographic Imager (casi): A push-broom VNIR hyper-spectral scanner which requires a smooth linear forward motion, normally that of an aircraft containing the instrument, to produce a two dimensional spectral image.

Forward Looking Infrared (FLIR): Passive infrared imagery obtained from cameras mounted on a mine-detecting vehicle.

Global Region Classification (GRC): A stage in the high level of the Remote Minefield Detection hierarchical algorithm that assigns the feature vectors to their most likely group.

Global Region Feature Extraction (GRFE): A stage in the high level of the Remote Minefield Detection hierarchical algorithm in which various morphological features such as pixel area, average pixel intensity and region compactness are measured for image regions.

Image Correction (IC): A stage in the Remote Minefield Detection hierarchy that compensates for distortions, dropouts, overlapping swaths, misregistration, and other artifacts and imperfections due to the scanning process.

Improved Landmine Detector Project (ILDP): A project to provide a remotely operated multi-sensor, vehicle mounted mine detection system to detect and mark low-metal content landmines

Local Region Classification (LRC): A stage in the low level of the Remote Minefield Detection hierarchy that assigns the feature vectors to their most likely group.

Local Region Feature Extraction (LRFE): A stage in the low level of the Remote Minefield Detection hierarchy in which various morphological features such as pixel area, average pixel intensity and region compactness are measured for image regions.

Local Region Assemble (LRASSEMBLE): An extension of LRT module to construct grey scale subimages around the suspect objects so that LRS can analyze.

Local Region Segmentation (LRS): A stage in the low level of the Remote Minefield Detection hierarchy, which segments the subimages formed by the LRT stage into regions

Local Region Thinning (LRT): A stage in the Remote Minefield Detection hierarchy that eliminates redundant subimages formed by the NSRR stage.

Non-Suspect Region Rejection (NSRR): A stage in the Remote Minefield Detection hierarchy which reduces the image data to small subimages that are likely candidates to contain mines.

Remote Minefield Detection (RMD): A process of detecting minefields from an airborne platform. RMD is also called Standoff Minefield Detection.

Remote Minefield Detection Hierarchy (RMDH): A scheme that includes a collection of algorithms for the purpose of detecting and identifying minefields from images.

Threat Detection Group (TDG): A research group at Defence R&D Canada – Suffield (DRDC Suffield), formerly known as Defence Research Establishment Suffield. TDG maintains an ongoing research and development program on the detection of unexploded ordnance such as landmines and artillery shells.

Target Spatial Analysis (TSA): A stage in the Remote Minefield Detection hierarchical algorithm that attempts to describe and interpret the relative positions of suspect mines.

1. Introduction

The Threat Detection Group (TDG) at Defence R&D Canada - Suffield has been investigating real-time methods of detecting minefields from airborne imaging platforms and close-in detection of mines since the mid 1970's.

The University of British Columbia (Principal Investigator: Professor Mabo Ito), has been collaborating under contract to DRDC Suffield, originally studying methods of processing airborne minefield images in real-time with an aim toward detecting surface-laid minefields. This led to the Remote Minefield Detection (RMD) hierarchical image analysis algorithm for single band (monochromatic) imagery that has the potential for real-time detection of minefields. More recently, this was expanded to include close-in detection of individual mines, such as those in Forward Looking Infrared (FLIR) imagery.

RMD Hierarchical Algorithm

The hierarchical algorithm, described in [1] and illustrated in Figure 2.1, going from a low to a high level of algorithm complexity, consists of the following levels:

- Preprocessing: image correction and sensor specific algorithm
- Low Level: preprocessing, target cueing (non suspect region rejection, local thinning)
- Mid Level target shape analysis (local region segmentation, local region feature, extraction, local region classification)
- High Level: target spatial analysis (clustering, syntactic and/or statistical feature extraction)
- Top Level: knowledge integration (spatial classification, data fusion)

Under the previous work contracts between DRDC Suffield and the University of British Columbia, the algorithm was completely specified and systems integration analysis, including analysis of data flow and message passing at all levels, has been done. The algorithm has been implemented in the Interactive Data Language (IDL) high-level language to allow rapid testing, modification and design improvements to be done. All portions of the algorithm, up to the high level, have been implemented as real-time code on the present target computer architecture, which is a scaled down version of what is necessary for a practical mine detection system. The algorithm has not, however, been run as a single coherent entity on a full-scale version of the architecture.

The current architecture, conceived in 1989, is based on a distributed network of transputer computing nodes, connected to a workstation. The low and mid levels of the algorithm are implemented on the network, while the high and top levels are implemented on the workstation.

Although the transputer was an attractive computing element, offering simple point-to-point serial communications and a high degree of scalability, the technology is now over 15 years old Although improvements have been made, in recent years the speed performance of an individual transputer has fallen badly behind other digital signal processor (DSP) boards, while the latter have improved both ease of inter-processor communication and price It is not, however, clear whether such DSPs would be suitable for the algorithm implementation. An assessment has been made comparing the different DSP, transputer and other available computing elements, in the context of the algorithm, to see which is best suited

An expert system was installed to handle the top level of the algorithm It is responsible for knowledge integration that includes the spatial classification and data fusion. The data fusion involves integrating image data with other non-image data, such as context or environment data A general testing methodology was defined to verify that the expert system and knowledge base are working correctly and the scope of the knowledge base was defined Due to urgent need, effort in the last two years was concentrated on adapting the algorithm for FLIR imagery from the Improved Land mine Detector Project (ILDP) vehicle This application did not require the high and top levels. Thus, the high and top levels have only been superficially developed and tested In particular, the knowledge base has only been implemented in a cursory manner. Detailed knowledge engineering, which includes developing the detailed knowledge base, implementing that knowledge base in the expert system shell and testing the complete expert system has not been done.

Initial research on the hierarchical algorithm was directed toward active airborne infrared imaging (AAII) However, the algorithm can be used for any single band imagery, and recent work has been investigating its application to forward looking infrared imagery from the ILDP vehicle. A suitable algorithm could assist or replace the present operator-in-the-loop. An IDL version has been developed but it has not yet been implemented in real-time. The resultant modified algorithm is similar to the original, except that the high and top levels are not needed and data rates are less stringent than for the AAII. Another potential application is for hyper-spectral images of surface-laid mines obtained from the Itres Research *casi* VNIR hyper-spectral imager, since these can be reduced to single band high contrast images that are very amenable to analysis by the algorithm. A preliminary data flow and architecture configuration analysis was completed. However, under contract to DRDC Suffield, the *casi* has been undergoing significant changes to its computer architecture during the last two years That and time constraints made it impossible to complete the development of the adapted algorithm, as planned

Research & Development Objectives and Phases

The objectives of this contract are:

- Implement and thoroughly test as a single coherent entity on a full-scale version of the current (transputer) architecture in real-time using FLIR imagery.

- Develop the high and top levels of the algorithm, in particular the expert system and knowledge base

- Explore new computational elements and architectures for implementation of the algorithm.

The 3-year contract was originally divided into three phases.

I) The first phase consists of the following tasks:

1. Carry out a trade off study of computational elements (processors), for use as nodes in a distributed computing architecture on which the algorithm could execute. The study should describe each processor, including the current one, and address, as a minimum

- Processor performance: discuss scalability, computational power, and inter-processor communication.

- Ease of software and hardware development: discuss what hardware and development software would be required. Discuss how easy it is to develop hardware systems and code.

- Adaptability: discuss overall suitability to the algorithm. Describe what modifications would have to be done to the existing algorithm and architecture (i.e , whether minor or major revisions or a total redesign is required).

- Cost effectiveness

2. Complete initial knowledge acquisition for high and top levels. This may involve extensive interviews with DND mine/countermine engineers who are experts in mine detection and/or military maneuvers. This may also require the contractor to employ a knowledge engineer as a consultant.

II) The second phase includes.

1. Implement the algorithm as a single coherent entity on a down scaled version of the current (transputer) architecture in real-time. Use the FLIR imagery as the initial case study.

2. Test the algorithm implementation on real-time FLIR imagery. Determine the maximum image rate that can be processed in real-time.

III) The third phase involves

1. Complete detailed knowledge engineering for the top level This will include the following subtasks:

- Develop the detailed knowledge base from information obtained in knowledge acquisition task.

- Implement that knowledge base in the expert system shell.

- Test the complete expert system.

2 Refine knowledge base as required based on testing with AAII

By the end of the first phase, two significant things had become clear that required Phase 2 to be modified:

- Although transputers were attractive computing elements 15 years ago when this project started, they had become obsolete Modern computing elements like Intel Pentium-class processors far outclassed them for processing power and cost.

- There was a lack of availability of tactical mine experts from Canadian Forces Engineers and Armoured Corps for the time necessary for, for the RMD expert system development

Thus, a decision was made by the Scientific Authority to implement the RMD hierarchical algorithm on the new hardware platform, which was recommended by a study from the University of British Columbia in Phase 1 Another decision was that the RMD expert system would be developed as an entry level shell only, and a future work will detail the knowledge and rule engines and any necessary facilities

Thus, the work in this contract was re-organized as:

- Phase 1 and 2 focused on the implementation of Low and Middle levels of a real-time RMD system on a transputer network to detect mines in FLIR imagery. The hardware requirement for each module of the algorithm was detailed, then the software development was outlined, and finally p some preliminary test results were presented.

- Phase 3 involved building a network of Intel Pentium-class processors, and developing and testing the new software version. An expert system shell was also developed in this phase.

2. RMD Hierarchical Algorithm and FLIR Imagery

This section briefly describes the Remote Minefield Detection Hierarchical (RMDH) algorithm, and introduces Forward Looking Infrared (FLIR) imagery of land mines.

2.1 RMD Hierarchical Algorithm

The structure of the Remote Minefield Detection algorithm for real-time detection of sparse small objects in images can be described using Figure 2.1. The major parts of the algorithm consist of (i) the Low Level Target Cueing to reject non-suspect regions and thus drastically reduce the data rate, (ii) the Middle Level Target Shape Analysis to classify the suspect regions as target or non-target relying upon their morphological features, (iii) the High Level Target Spatial Analysis to extract the features of the spatial relationship between mine-like objects, and (iv) the Top Level Knowledge Integration to resolve whether or not the image contains mines using the spatial analysis results and external information resources.

Raw image data is acquired by a sensor and passed into the Image Correction (IC) stage. This IC module adjusts the raw image data to compensate for distortions, dropouts, overlapping swaths, misregistration, and other artifacts and imperfections due to the scanning process. After the IC stage, the data is passed to the Non-Suspect Region Rejection (NSRR) stage. The NSRR reduces the immense data flow down to a stream of small images (subimages) that are likely candidates to contain mines. To accomplish this task, the NSRR collects a block of scan lines of the image and then divides it into smaller non-overlapping square regions called subimages. Each subimage is further divided into smaller non-overlapping square regions, whose width is smaller than the dimensions of a small target but large enough to provide a stable average of the region. The contrast values of these small regions against the parent subimage are calculated and ranked. Only a few top values will be selected as suspect regions. Then the data of these suspect regions is transferred to the Local Region Thinning (LRT) module where redundant subimages are eliminated. Within this module, function Local Region Assemble (LRASSEMBLE) constructs subimages around the suspect objects. The next stage, Local Region Segmentation (LRS), partitions these subimages into homogeneous regions. These regions are then passed to the Local Region Feature Extraction (LRFE) block. In this step, various morphological features such as pixel area, average pixel intensity and region compactness are measured for each candidate mine-like object. The next stage, Local Region Classification (LRC), categorizes the assembled feature vectors into different classes, to determine targets or non-targets with an estimated likelihood based on extracted features. The results of the classification are passed to the higher level, Target Spatial Analysis. Here, initially the relative positions of likely mines are analyzed and various spatial size and shape clusters are formed using a Clustering algorithm. Then, the clusters are delivered to Global Region Feature Extraction (GRFE) where both statistical measurements and pattern descriptors of each cluster, as well as spatial inter-relationship among clusters, are computed and extracted. Depending on the type of patterns encountered, either a statistical or a syntactic pattern classifier may be more appropriate. The positions of mines in scattered minefields can be well described by probability density functions

and hence are amenable to statistical classification The positions of mines in patterned minefields are better described by deterministic functions, which suggests syntactic classification However, since the patterns are not known in advance, both classifiers are needed and operate in parallel. In the final level, Knowledge Integration, the expert system integrates the statistical and syntactical data output from GRFE with other sources from the user and knowledge base to arrive at a decision on whether or not the raw image data likely contains a minefield

This system architecture was initially designed for AAII airborne imagery. Some modifications of the algorithm and hardware structure are being investigated to accommodate FLIR imagery

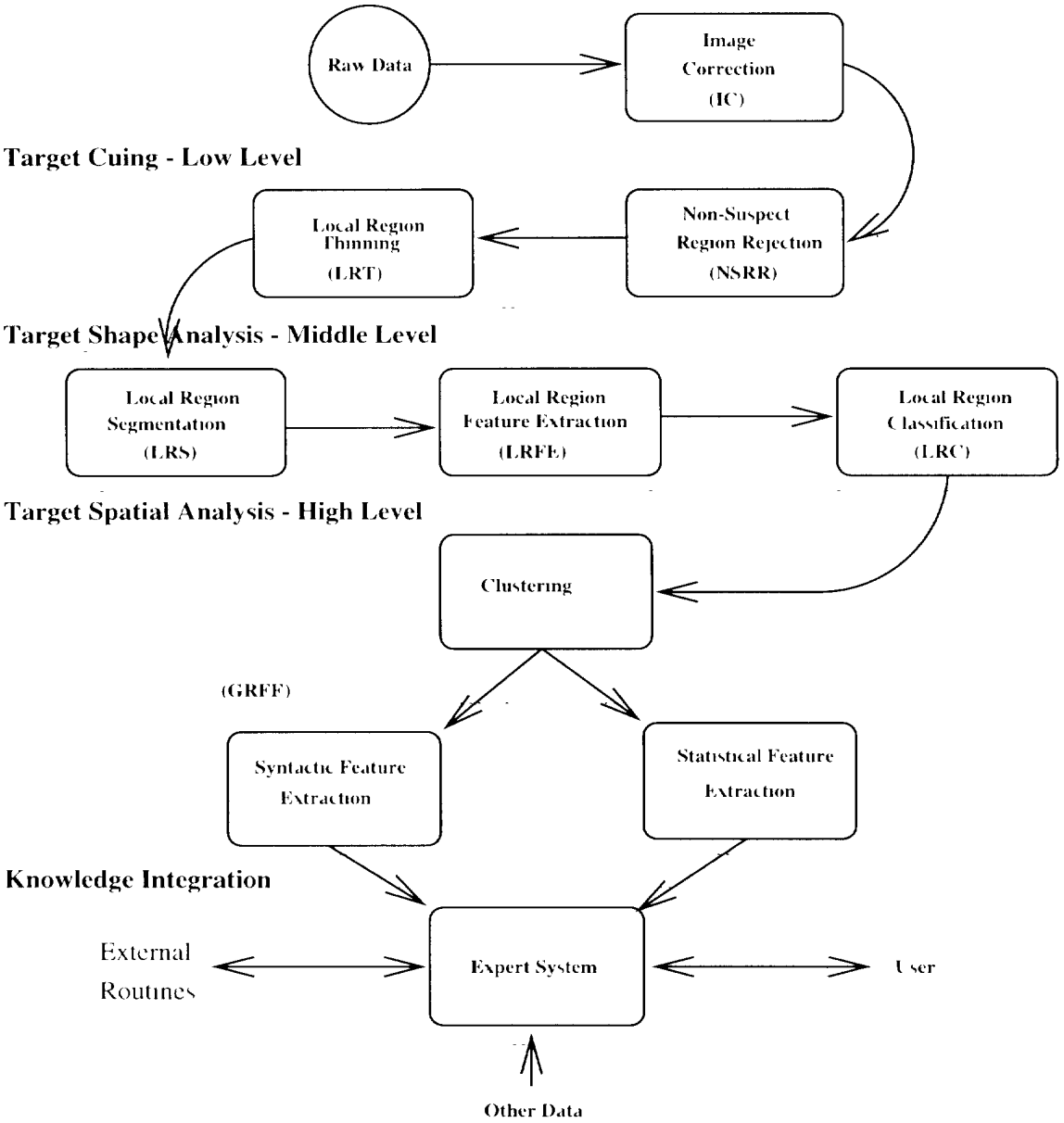


Figure 2.1: Structure of the Remote Minefield Detection hierarchical algorithm

2.2 FLIR Imagery of Land Mines

Land mines are often cylindrical but may be a variety of shapes. Diameters range typically from 10 to 30 cm for anti-personnel and anti-tank mines, respectively. Minefields may consist of buried or surface-laid mines. A typical minefield distribution might consist of over 200 mines in a 500 m square area.

A number of imaging detectors are being developed for use in remote minefield detection from airborne platforms and from slow moving ground vehicles. In this study, we focus on FLIR imagery from a multi-sensor remote mine detection vehicle that operates primarily on roads at slow speed about 3 6 km/h. A typical data rate for this imagery is 4 images/s or 1600 lines/s An image has 400 lines, each of which contains 800 16-bit samples. Due to the range dependent perspective, the size of mines depends on their position in the frame. In mid frame range, an anti-tank mine may be 32 - 40 pixels in diameter and an anti-personnel may be 12 - 20 pixels in diameter, along track and across track respectively. Typical expected signal to noise ratios are in the range 10 to 50.

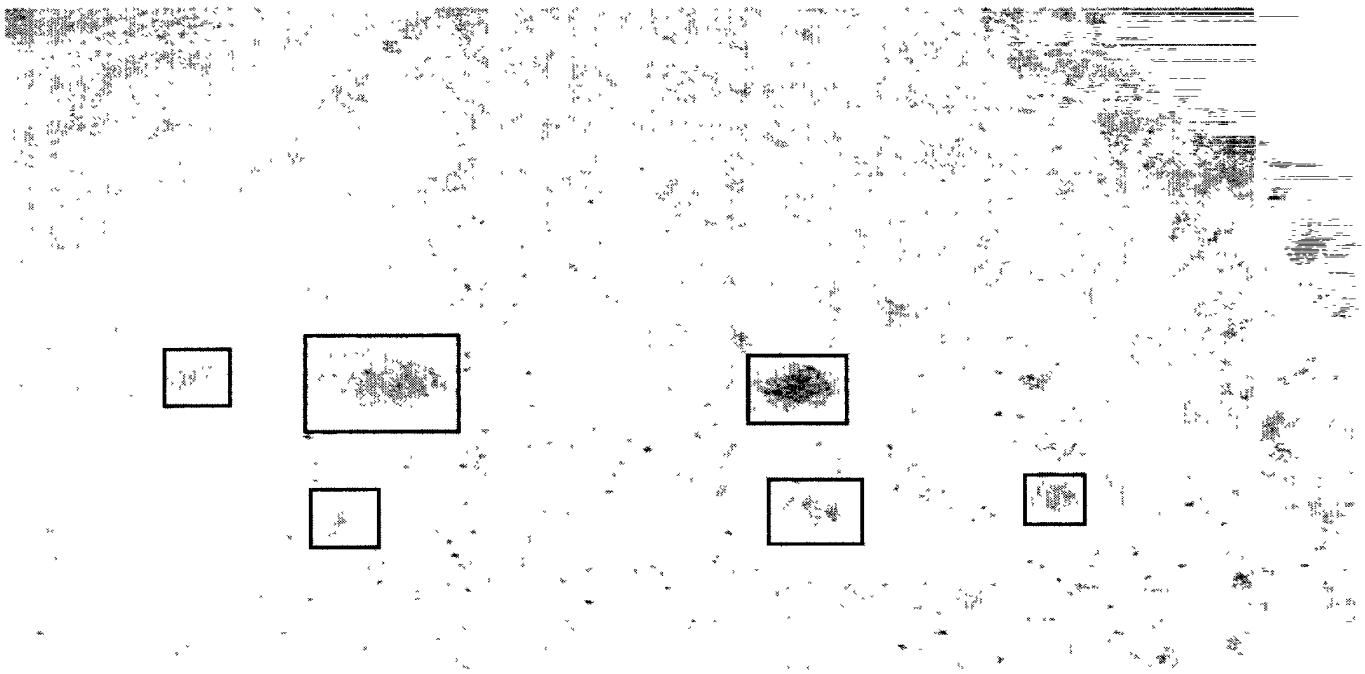


Figure 2.2: A FLIR image with 2 anti-tank and 4 anti-personnel mines visible

Because FLIR images are obtained from ground-based vehicles that are intended to detect individual mines, there is no spatial analysis required to detect a whole minefield. FLIR imagery characteristics and the RMD algorithm modification for this type of imagery are discussed in the

previous report "Implementing and Testing a Real-Time Hierarchical Mine Detection Algorithm" [19].

2.3 System Integration of RMD Algorithm for FLIR

To resolve the difficulties in communication between various modules and to determine the optimal hardware configuration, it is essential to study the systems integration of the RMD algorithm modules.

Following is the calculation of data flow in each module of the RMD system, based on the actual data input/output:

- ***NSRR**: detect the highest intensity contrasts (hits)*

input Data=flt(800, 400)
*output: Result=flt(4, Iclip * 6)· (x, y, index, max_intensity) of "Iclip" subimages in each of 6 scan lines of 800x50*

*Input data flow· 4 frames/s * (800*400 pixels/frame) * 2 bytes/pixel = 2.5 Mbytes/s*
*Output data flow. 4 frames/s * 4 data/iclip * 2 bytes/data * 32 subimages/scanlines * 6 scanlines/frame = 6 kbytes/s*
*or, 4 frames/s * 32 subimages/scanlines * 6 scanlines/frame = 768 subimages/s*

- ***LRT**: make all the intensity contrasts of one hit become unique*

*input=· Result=flt(4, Iclip * 6) ;Iclip=32*
output. Hits=flt(4, number_of_hits) ;number_of_hits = ~40 hits/frame

Input data flow: 6 kbytes/s from NSRR output
*Output data flow. 4 frames/s * 4 data/hit * 2 bytes/data * ~40 hits/frame = ~1 25 kbytes/s*

These data flows are low comparing with those in NSRR Therefore, the data flows in NSRR are dominant.

- ***LRA**: reconstruct the shape of the unique hits in the database*

*input: Data=flt(800, 400), Result=flt(4, Iclip * 6)*
output: Grey=flt(marksize, marksize, number_of_hits) ;marksize=64 pixels

Input data flow about the same with that in NSRR, ei. 2 5 Mbytes/s
*Output data flow. 4 frames/s * (64*64 pixels) * 2 bytes/pixel * ~40 hits/frame = ~1.25 Mbytes/s*

Therefore, the data flow of 2.5 Mbytes/s is dominant

- ***LRS** (RSAM). segmentation*

input: Grey=flt(marksize, marksize, number_of_hits) ,marksize=64
output: Segmented=byt(marksize, marksize, number_of_hits)

Input data flow: ~1.25 Mbytes/s as in LRA's output data flow
*Output data flow: 4 frames/s * (64*64 data) * 1 bytes/data * ~40 hits/frame = ~640 kbytes/s*

- ***LRFE**: Feature extraction*

input: Segmented=byt(marksize, marksize, number_of_hits), Grey=flt(marksize, marksize, number_of_hits)
*output: features=flt(13, 2 * number_of_hits)*

Input data flow: ~640 kbytes/s + ~1.25 Mbytes/s = ~1.9 Mbytes/s
*Output data flow: 4 frames/s * (13 * 2 * ~40 hits/frame) * 2 bytes/hit = ~8 1 kbytes/s*

- ***LRC**: Classification*

*input: features=flt(13, 2 * number_of_hits), Design=flt(11, 12, 2)*
*output: Classified=flt(6, 2 * number_of_hits)*

Input data flow: ~8.1 kbytes/s from LRFE
*Output data flow: 4 frames/s * (6 * 2 * ~40) data/frame * 2 bytes/data = ~3.75 kbytes/s*

The systems integration of the RMD algorithm performed on FLIR is summarized in Figure 2.3, which shows the data flow in the levels.

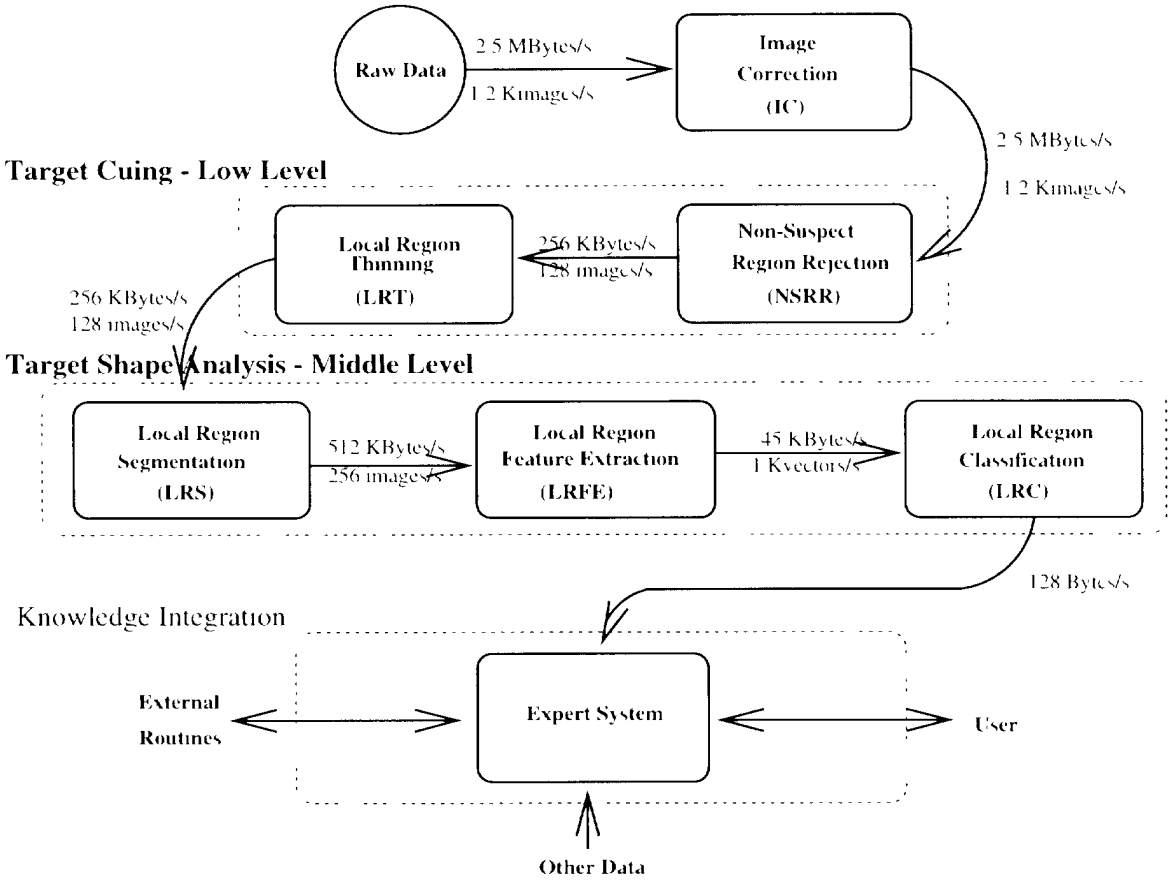


Figure 2.3: Systems integration analysis for FLIR imagery.

2.4 A Search for a Better Segmentation Algorithm

The currently used segmentation algorithm is based on the split-and-merge technique by the Horowitz & Pavlidis, codes developed by AIST MITI (JAPAN) 1983. This algorithm was chosen because of its availability at the time and because it yielded satisfactory results. As stated in section 4 2.3 of the "Implementing and Testing a Real-Time Hierarchical Mine Detection Algorithm – Final Report" [19], the current segmentation algorithm RSAM sometimes fails to detect the true shapes of mines due to small "bridges" which connect the mine images with surrounding debris (Fig 2.4)

Scanned images of anti-tank mines Scanned images of anti-personnel mines
(In each case, a good image is at left, and a bad image with narrow "bridges" is at right)

Figure 2.4: Mine images detected by RSAM segmentation algorithm.

As an attempt to improve the result, a number of segmentation algorithms have been examined and tested using FLIR images. A number of routines for image segmentation from the University of Kansas KUIM Image Processing System were tested on our FLIR imagery. Refer to http://www.ittc.ukans.edu/~jgauch/research/kuim/html/07.00.html for details of each algorithm. The summary descriptions from that Web Site are included below

- **Intensity Thresholding**

When the intensity histogram of an image is bimodal, it is often possible to identify objects of interest in an image by thresholding. All pixels greater than the specified threshold value are part of the object and labeled with 1's, and all pixels less than the threshold value are part of the background and assigned 0's The *thresh* program has a number of options which let the user specify the threshold, or choose an automatic threshold based on edge strengths (-e) or histogram statistics.

- **Region Growing**

The *region_grow* program implements seed based region growing for image segmentation. A region is detected by identifying the set of points that are connected to the seed point that are "similar" in value A recursive algorithm is used to grow a region whose pixels are in the range [m-t .. m+t] where m is the mean intensity of the region so far, and t is a user supplied threshold. The -s switch is used to specify the starting point, and the output of the program is a binary image (1's = object, 0's = background).

- **Split Merge**

The *split_merge* program performs the classic recursive split and merge segmentation algorithm. The program begins by breaking the image into a large number of homogeneous regions, and then combines adjacent regions to eliminate artificial boundaries caused by the recursive splitting process. Both the splitting and merging processes use intensity variance in a region as a measure of its homogeneity A user specified variance threshold indirectly controls the number and size of regions produced by this algorithm. The output image contains the average intensity value for each of the regions identified.

- **Boundary Detection**

The *bound* program reads in a segmentation image (either B/W or multi-valued) and uses a top to bottom, left to right scan of the image to locate region boundaries. Unlike the *contour* program, these boundaries are not "tracked" or output to an ASCII file. For display purposes it is often useful to superimpose these boundaries to the original un-segmented image to demonstrate the quality of segmentation results

- **Contour Following**

Once an object of interest has been identified by thresholding or another segmentation method, it is often necessary to find the sequence of points on the object boundary. The *contour* program takes a B/W image and tracks the boundaries of each object and writes the (x,y) coordinates to an ASCII file for future use. At the same time a B/W image the same size as the input image is created which shows the contour points

- **Blob Coloring**

Once an image has been segmented into as number of regions using thresholding or another method, it is often useful to identify each of the connected components in the image. The process of assigning each distinct region unique identifiers is called blob coloring The *blob_color* program uses a simple recursive search algorithm to find all points connected a starting point. Several program options enable the user to search for white objects on black backgrounds or *vice versa,* and vary the definition of "connected" to use 4 neighbors or 8 neighbors

- **Maximum Likelihood Segmentation**

The *ml_segment* program performs maximum likelihood segmentation on scalar and vector valued images. The process is similar in spirit to the "quantize" program except that the intensity distribution of pixels in each "segment" are explicitly modeled.

- **Gradient Watershed Regions**

The *water_merge* program calculates gradient watershed regions and then performs an iterative merging procedure to combine these regions based on their "similarity" The result is a segmentation image with very good edge localization

- **Intensity Watershed Regions**

The *watershed* program partitions an image into watershed regions by following the intensity function "downhill" to the nearest intensity minima. Regions associated with intensity maxima can also be calculated. Regions associated with watersheds of the gradient magnitude of the input image can also be calculated

- **Crack Detection**

The *crack* program searches an image for dark thin cracks. This is done by scanning the image left-right, top-bottom and diagonally at each point looking for large increases in intensity.

Some preliminary tests were conducted FLIR images were converted to Graphics Interchange Format (GIF) and tested with the above stand-alone segmentation algorithms. Although the computation was complex, a few of the tested algorithms could separate a few of the clusters that RSAM failed. Thus they should yield slightly more accurate RMD results. It was concluded that Intensity Thresholding, Region Growing, and Contour Following were possible alternatives to the current RSAM algorithm. However, further tests should be conducted to confirm this conclusion.

As another approach, one can improve the RSAM results, such as to eliminate those "small bridges", by a routine that recognizes a mine-debris coupling and then automatically adjusts the RSAM parameters accordingly. Furthermore, methods from mathematical morphology, such as dilation and contraction operations, are often used to remove small islands and connecting ridges from images. Future studies should examine the ability of mathematical morphology techniques to eliminate the bridges in small AP mine images.

3. Hardware Configuration for the Real-Time RMD System

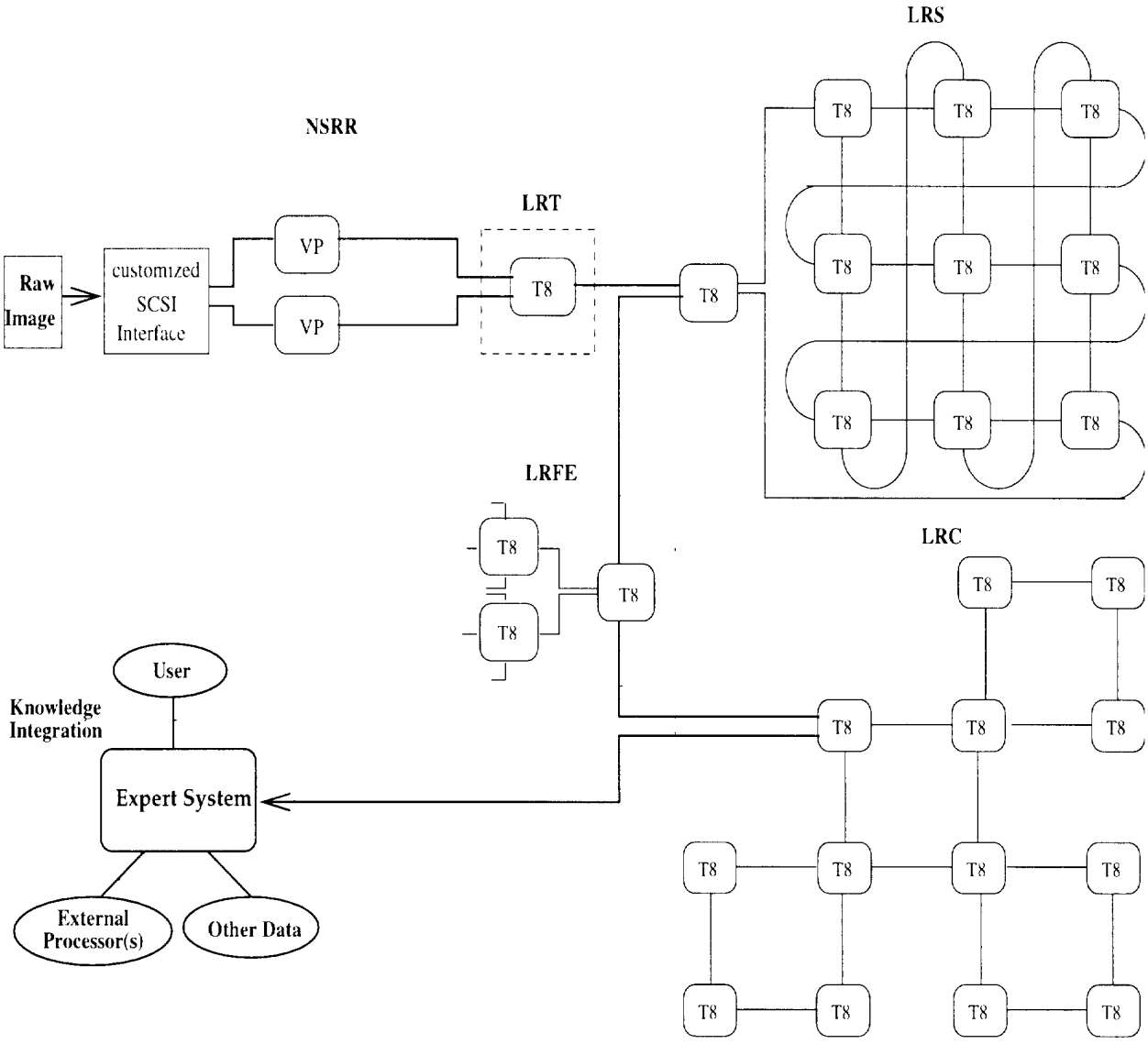
3.1 Implementation Using Transputers

In order to achieve real-time operation, a distributed computer system is essential.[3, 16] This system employs a large number of processors with varying topologies for each of the processing nodes. Thus it uses a simple point-to-point communications protocol for interconnection. The network is scaleable, i.e., increasing the number of elements requires only minor changes to the programming code.

Using a cost tradeoff and data rate analysis, a parallel processing computer hardware configuration proposal for a FLIR minefield detection system is illustrated in Figure 3.1. This system utilizes an array of T800 transputers (T8), i860 vector processors (VP), and a Sparc station for operator interface and storage.

Computation of processing time for NSRR, LRT and LRFE modules based on transputer data sheets is described in Appendix A. This calculation suggests a total of 40 T800 or 9 T9000 transputers are needed for NSRR, and 1 T805 30 MHz for LRT. However, to solve the i/o bottleneck problem between the SCSI bus and the RMD system input, 2 TTM110 vector processors (which have a throughput of 80 MFlops/s, almost four times the throughput of T9000 yet simpler to code) are used in the NSRR, and a number of T405 transputers might be employed for a customized SCSI interface.[16] For LRFE, 3 T805 30 MHz transputers would be sufficient to handle the load, these T800s communicate among themselves using process farm configuration.

However, today transputer technology is quite old and is behind many modern computational elements, such as those found in DSP boards or general purposed processors, in terms of features, ease of use and price/performance ratio. Transputer production has been halted by the manufacturer (INMOS), who no longer provide product or development tool support. Thus, transputer networks are becoming increasingly difficult to support and maintain. The current distributed computing architecture based on transputers will need to be upgraded in the future and thus a network of Pentium-class PCs was build to test the RMD algorithm in Phase 3.



VP: i860 based vector processor. T8: T805-30 transputer with local memory.

Figure 3.1: Hardware configuration proposal for the real-time RMD system.

Figures 3.2 and 3.3 show the hardware configurations for the NSRR using T9000 and T800 transputers, and Figure 3.4 shows the hardware configuration for the LRT using one T800:

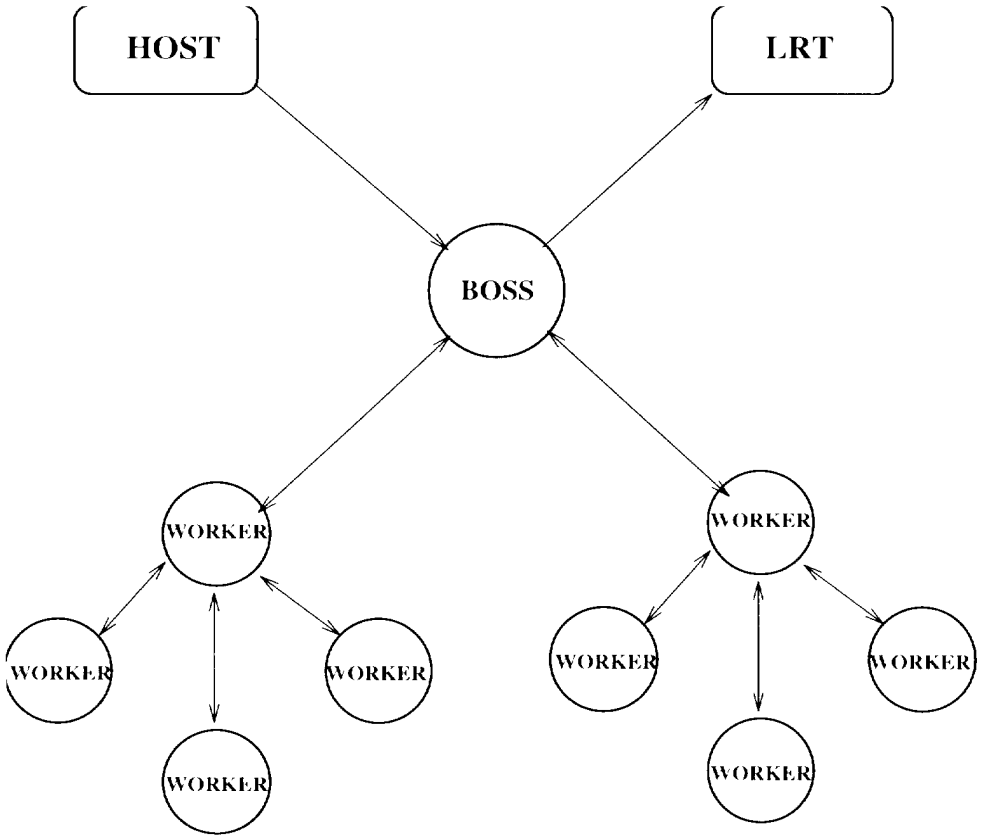


Figure 3.2: Proposed computing architecture for the NSRR using 9 T9000 transputers.

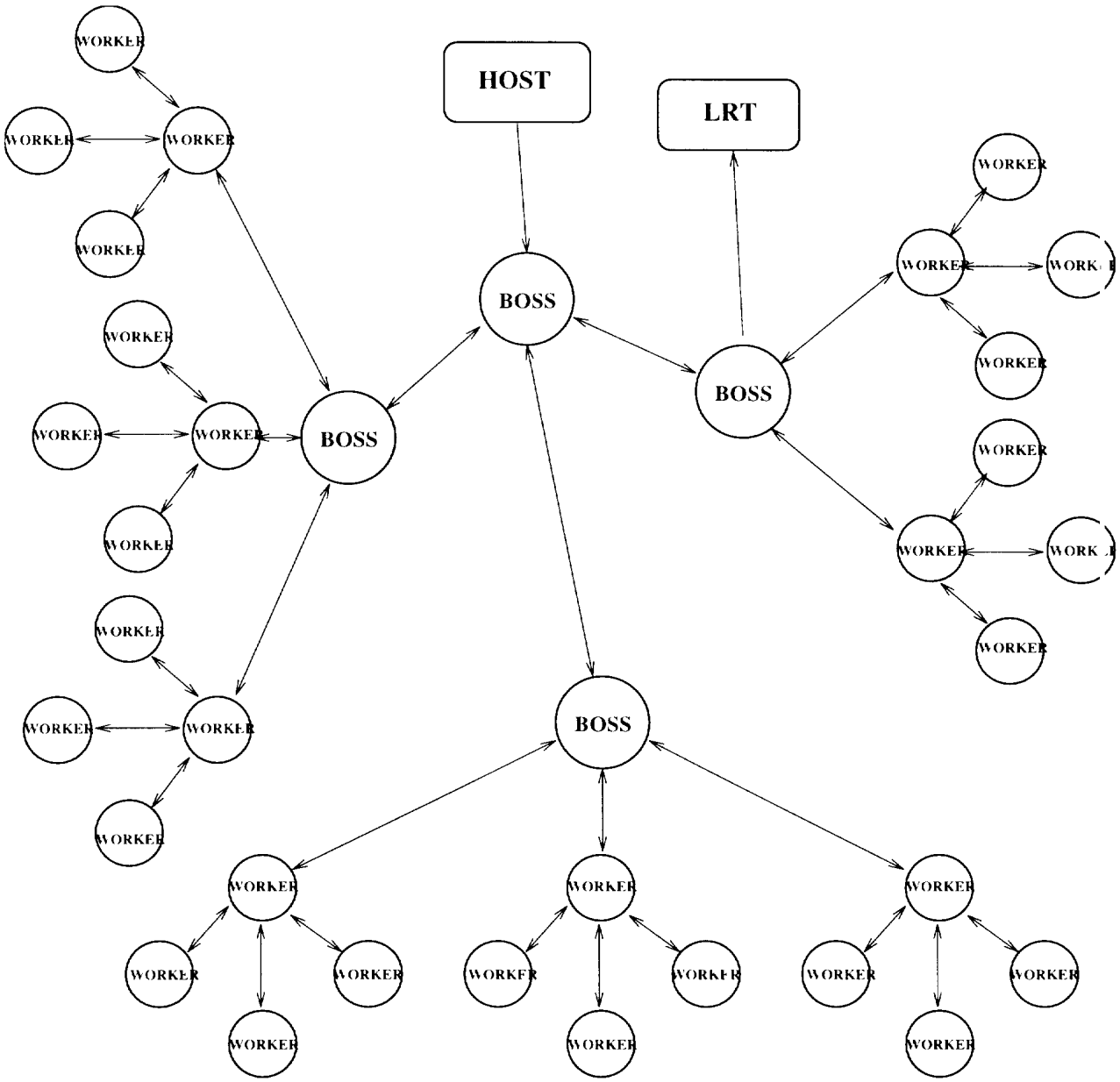


Figure 3.3: Proposed computing architecture for the NSRR using 40 T800 transputers.

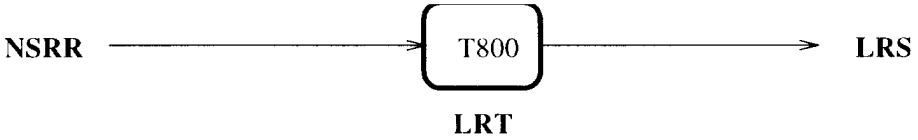


Figure 3.4: Proposed computing architecture for the LRT using 1 T800 transputer.

To form the above configurations, the transputers are hardwired using a 4x3 mesh structure as shown in Figure 3.5, and then software is used to configure the data/control interconnections.

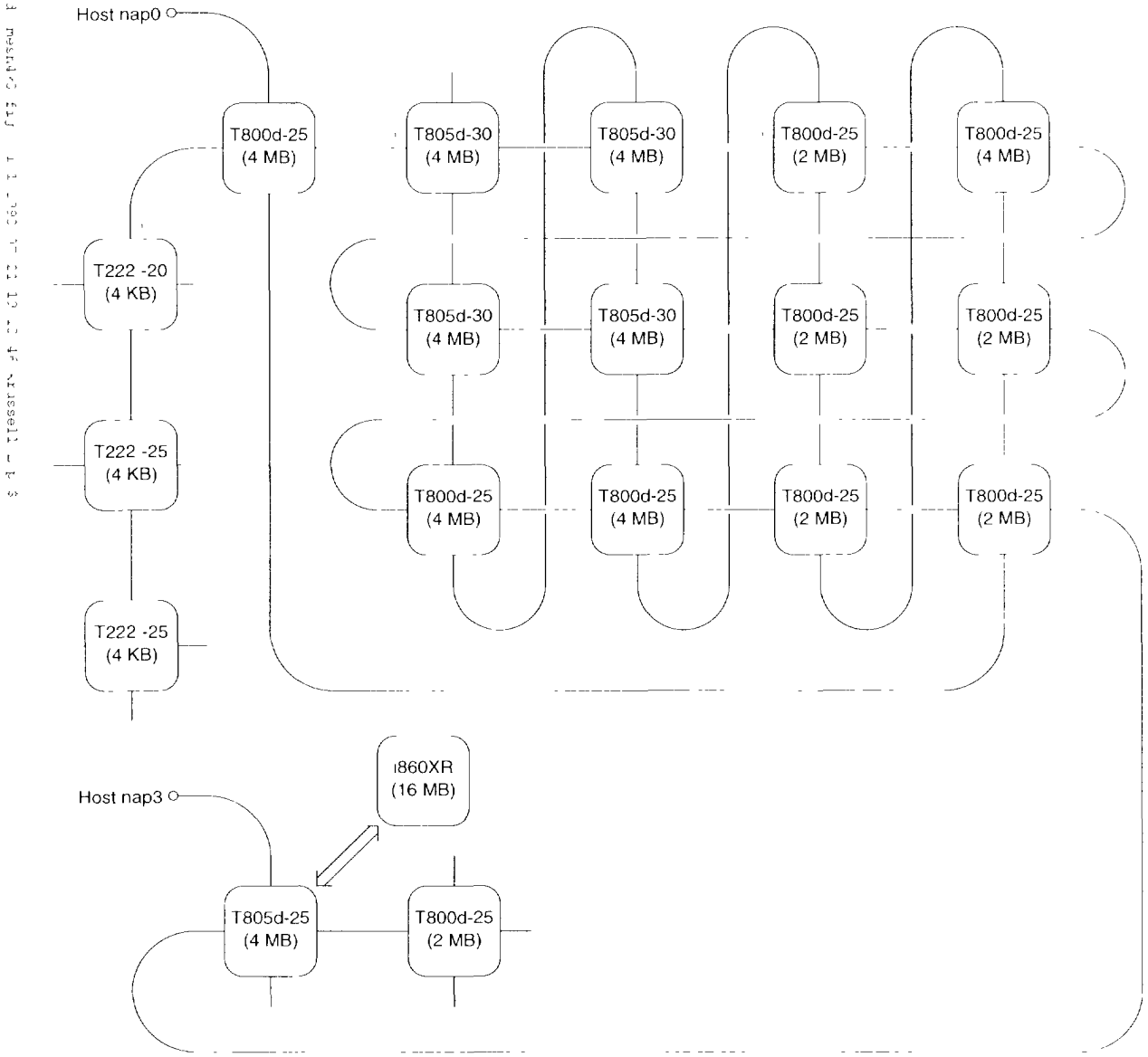


Figure 3.5: Transputers hardwired in a 4x3 mesh structure.

3.2 Alternative Hardware Implementation

Although transputers were attractive computing elements 15 years ago when the RMD project started, they have become obsolete. Therefore, a study of an alternative hardware implementation of the Remote Minefield Detection system had been conducted by the University of British Columbia [22] in Phase 1 of the contract.

This study concludes, "The RMD system performs substantial image processing, but does not appear to fully utilize the enhanced abilities of a digital signal processor. Thus, it is not recommended that a digital signal processor such as the SHARC be utilized. Instead general-purpose processors such as the PowerPC, Pentium and Alpha should be considered. The generation of these processors should be picked such that those that have SIMD capabilities are utilized as appropriate. Usually, it is only possible to obtain SIMD general-purpose processors easily, so this is not a difficult proposition. Besides being much faster at the image analysis than the SHARC, these general-purpose processors have the advantage of being much cheaper, as well as having much lower development costs. The SHARC is more optimized for execution of fast multiplies and accumulates, but this is not something that the RMD system does much of. Additionally, much of the speed of the SHARC is based on the three independent execution units and register file - ideally the SHARC would have two execution units executing at once, but that was not the case in the compiled code. Thus, the SHARC was essentially being used as a micro-controller, which a digital signal processor is not extremely good at (and it wastes resources available to the processor). NSRR and LRT are the two most computationally intensive functions that take a lot of hardware to run (taking 4 times as long when the image size doubles in both dimensions), while the remaining algorithms, LRA, RSAM and LRFE scale linearly as the image size increases."

This study also suggests that to improve the run-time speed, one should carefully choose a compiler based on:

- Compiler efficiency: how well the compiler can produce code.

- Compiler differences: how well each compiler can use the functions given (i.e., how much code has to be written because the compiler library call doesn't implement it), and the operating system.

Many compilers available in the market are for general-purpose processors, while only a few are for specialized processors (like the SHARC). Thus the compiler in the latter case is the weakest link in choosing a platform. Moreover, while assembly language development is possible, it is often desired to produce code in C. Then, the lack of additional compilers is often a drawback, since one is tied to a certain vendor who supplies the said compilers as well as the development tools.

Furthermore, general-purpose processors are likely lower in cost due to their widespread availability, so one is able to keep a few spares on hand. Also, documentation for general-purpose processor platforms is abundant and easy to find.

3.3 Implementation Using Intel Pentium Processors

In Phase 3, a new hardware platform for the RMD algorithm was built using:

- A workstation PC: Pentium IV 1 8 GHz, 256 MB SDRAM, 80 GB hard drive, 1GB Network Interface Card (NIC), MS Windows 2000. This workstation was intended for the new software version development, and for running the main module of the RMD program.

- 4 substation PCs: Pentium IV 1 6 GHz, 128 MB SDRAM, 20 GB hard drive, 1GB NIC, MS Windows XP A KVM switch was used to share a common monitor and a keyboard among these substations. These substations served as additional data processing units, which were controlled by the workstation

- A fast 1GB ethernet switch to connect these five PCs to form a private LAN. Each station was assigned a unique IP address, and communicated among themselves using TCP/IP protocol provided by the running Microsoft Windows OS. Only the workstation could connect to the outside infrastructure networks or internet using an additional NIC.

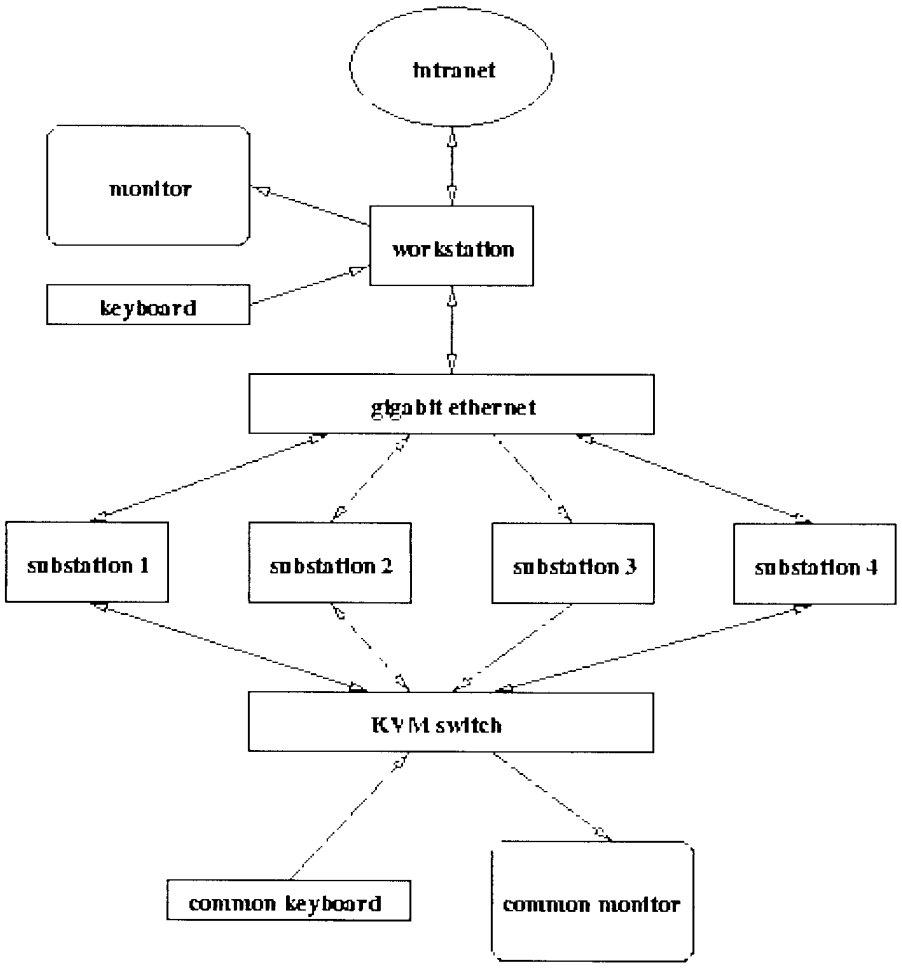


Figure 3.6: Diagram of the network of Pentium PCs

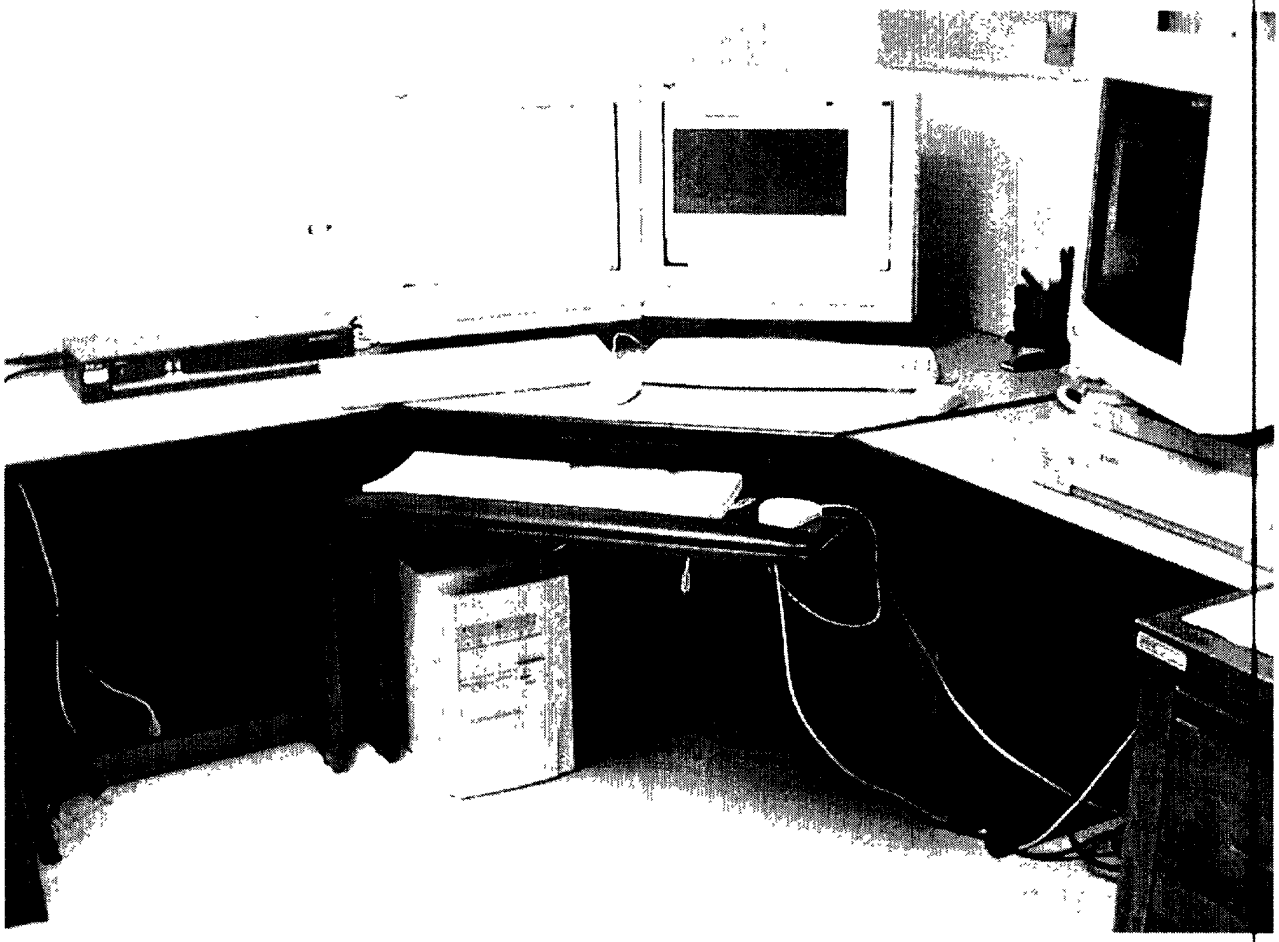


Figure 3.7: The workstation PC (under table) and the KVM switch (to the left of the left-hand monitor) with monitor and keyboard.

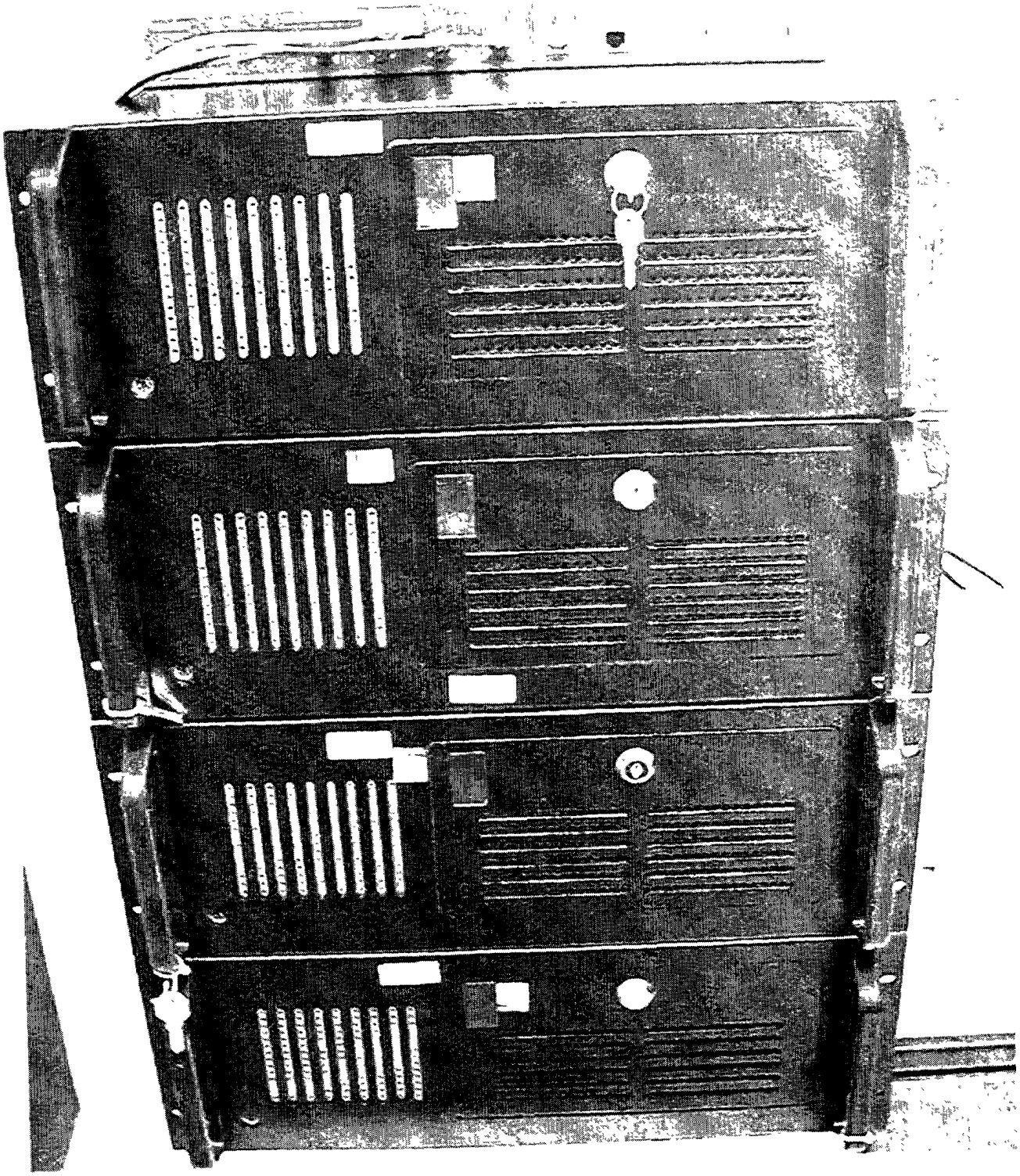


Figure 3.8: The substation PCs, with the gigabit ethernet switch on top

4. Software Development

4.1 Software for Transputer Network

Discussion of software development of a complete RMD algorithm is given in detail in the report "Image Analysis Application Development" [2]. Also, the communication protocol between the modules is described in the "Systems Integration Study of a Hierarchical Minefield Image Analysis Algorithm" report [16].

In the previous work contract, a simulation version of the low and middle levels of the hierarchy adapted for FLIR imagery was first programmed to run on a Unix-Sparc machine. The modifications from the original algorithm made for FLIR imagery are outlined in the report "Implementing and Testing a Real-Time Hierarchical Mine Detection Algorithm" [19].

After verifying the non real-time program running as intended, the code of this low and middle level of RMD hierarchical algorithm for FLIR was transferred to implement on a real-time version of an architecture of i860 vector processors and transputer TRAMs. This software was written in INMOS ANSI C and OCCAM (a common language for concurrent computing of transputers) and was run in Unix IDL environment for graphic user interface.

The source codes of the real-time version of RMD algorithm for FLIR imagery are stored in the TDG directory:

/net/frodo/export/home/projects/rmd/hieralgorithm/realtime-flir

The FLIR minefield images used for testing this real-time version of the RMD algorithm are stored in directory:

/net/frodo/export/home/projects/rmd/images/flir

As a new hardware platform will replace the transputer network in Phase 3, software development for the current transputer architecture ceased after the completion of LRFE.

4.2 Software for PC Network

Because of the time shortage for software development in Phase 3, only the NSRR, LRS (or RSAM), LRFE and LRC modules were coded in C and for running in DOS mode. The Interactive Data Language (IDL) package was needed to provide GUI services such as user interaction and graphic display. This arrangement, much like the simulation version running on Unix in previous contracts, was necessary for this early stage of software development. It provided quick debugging steps and easy integration for the modules under development. A simple DOS or windows application can easily replace this IDL environment in the future.

At first, it was planned to code the modules in C++, which utilizes Microsoft Foundation Class (MFC), true windows GUI, and multi-thread programming. This attempt was not completed after a trial period because of the short time budget for software development in Phase 3. Another original intent was to use the substation PCs to share the workload with the main workstation (Figure 4.2). Only preliminary work was done on this aspect, because the difficulties in inter-processor communication could not be fully resolved in the time available.

One can, however, improve the speed of the existing RMD algorithm implemented on a single PC by doing the following things:

- Store and share intermediate data arrays using RAM instead of system files. As Figure 4.1 shows, data files currently were retrieved and stored many times during the RMD process, thus it significantly slowed down the operation. This modification is one of the major factors to speed up the process, yet can be easily implemented.

- Use a faster single processor PC (2GHz or better), or a multi-processor shared-memory PC Moreover, a single PC model could be more preferable than a network of PCs, because it can avoid the potential LAN communication problem, reduce the cost of equipment and programming effort, and decrease the space occupied by the system.

- Optimize the codes, trim off debugging checkpoints, simplify the procedures, reduce the steps to read in and check for system call parameters and data formats (in fact, most of these parameters and data formats were fixed)

- Avoid passing data arrays stored in system files among substations in client/server fashion as this might increase considerably the overall processing time

If these modifications cannot make a single PC satisfy the real time speed requirement, the multi-PC model might become an option

As an illustration, the real time input speed of FLIR imagery is 4 frames / second, or 0.25 second for 1 frame. The current RMD software running on a single Pentium IV 1.8 GHz, with IDL service, a DOS shell and lot of system file swapping, could analyze a frame typically in 1.18 second. With the suggestions listed above, the software could possibly achieve 2 frames/second or even meet the required real time speed.

The source codes of the PC version of the RMD algorithm for FLIR imagery were compiled using Microsoft Visual C++ 6.0, and are archived in the TDG directory:

/net/frodo/export/home/projects/rmd/hieralgorithm/PC-FLIR

Figure 4.1: Data files are retrieved and stored during the RMD process

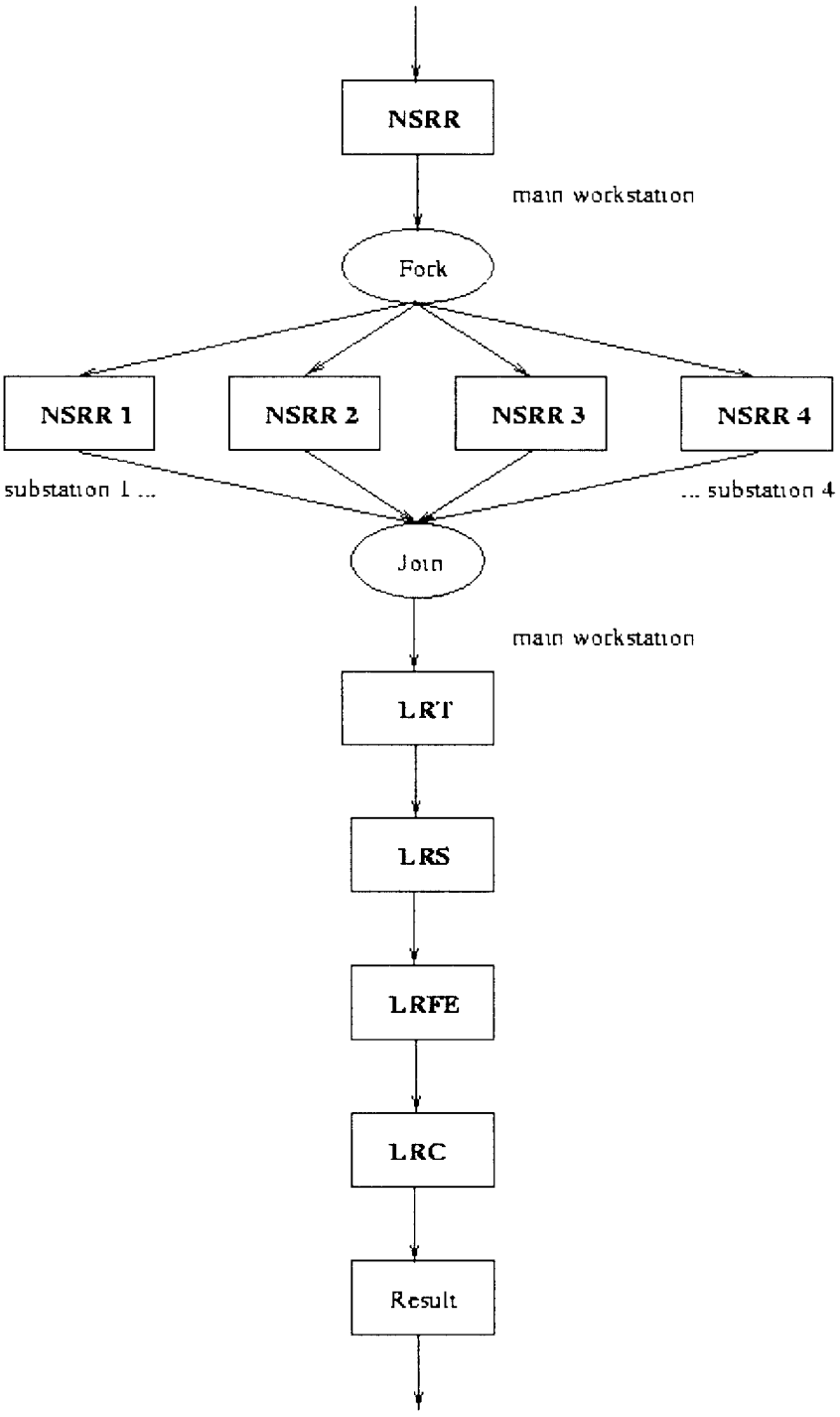


Figure 4.2: Proposed multi-PC function distribution. Only Fork and Join blocks are shown here for NSRR, but the same mechanism would be applied for LRS and LRFE as well.

4.3 Expert System Shell

An expert system for the top level of RMD algorithm was also developed in this contract. Because of the lack of availability of mine experts from the Canadian Forces Engineers and Armoured Corps during the time period required to fulfill this contract, only an entry level class of expert system shell was accomplished. This shell merely contained a simple knowledge base with a few facts and rules, and a query engine. Future work will be needed to enrich the knowledge base and to detail any necessary facilities. A discussion of the implementation of this expert system is given in Appendix A.

5. Preliminary Test Results

5.1 Test on Transputer Network

By the end of Phase 2 of the contract, only the codes for NSRR, LRT and LRFE were developed and tested separately. Then the RMD implementing on the network of transputers was ceased, and replaced by the development on the new multi Pentium PC platform. Note that as in the original plan, all the RMD modules should be integrated and run as a single coherent entity on a full scale version of the transputer architecture.

Real FLIR images, acquired from ILDP, were stored in AGEMA's image format and were transformed into TDG format (a custom format) using IDL routines. Each image frame has 800 x 400 x 12-bit pixels. The data input rate for the real-time version is expected to be 4 frames/s, which yields the data reading time of 0.25 seconds/frame.

Figure 5.1 shows a typical processing time for NSRR, LRT and LRFE modules in the real-time version of the algorithm. Although these speeds are quite fast in comparison to those in the simulation version, they are not fast enough for the real time frame rate requirement of 0.25 seconds/frame. Furthermore, the data results from the real time version were verified against the results from the simulation to make sure they are the same.

Module	Real-time	Simulation
NSRR (2 TTM110)	0 118272	0.25207007
LRT (1 T805-30)	0.129952	0.31768095
LRFE (3 T805-30)	0.257400	0.69026039

Figure 5.1: Processing time (seconds) for Real-Time versus Simulation version of the algorithm

5.2 Test on PC Network

Using Interactive Data Language (IDL) package to provide GUI services and simple integration the NSRR, LRS (or RSAM), LRFE and LRC modules (which were coded in C and run in DOS mode) were tested along with the LRT and LRA (which were coded in IDL language). A particular test performed on the single workstation PC yielded the processing time as recorded in Figure 5.2.

As shown in this table, the total processing time for a FLIR image is typically 1.18 second on the Pentium IV 1.8 GHz machine. This is quite fast in comparison with the simulation version on the Unix SPARCstation-20 platform, but the real time speed required for FLIR imagery is 4 frames/second or 0.25 second per one frame. Note also that the slowest module is probably the LRS, which could be improved by employing a faster segmentation algorithm, giving the same result.

As mentioned in section 4.2 (Software for PC Network), the processing speed can possibly be made much faster by eliminating the file swapping and by optimizing the existing codes. Furthermore, more R&D work will be needed to implement the multi-PC network as originally planned, and thus to achieve the real time speed requirement.

Machine	NSRR	LRT	LRA	LRS	LRFE	LRC	Total
P4 1.8GHz	0159	015	.016	437	375	.156	1 18
Sparc 20	.114	144	.139	4.36	7.53	2.35	16 21

Figure 5.2: Processing time (seconds) for Pentium PC versus Unix Simulation version of the algorithm

5.2 Rate of Success

At the time there were 44 FLIR images available for testing. Eleven of them were so poor in quality that they cannot be used to detect mines even by visual inspection. The FLIR algorithm was run with the remaining images and the following results were obtained:

Image	T.P.A.*	Image	T.P.A.*
960716_000000_0170	2 1 4	960716_041500_0160	2 3 5
960716_001500_0359	2 3 6	960716_043000_0370	2 3 5
960716_003000_0110	2 1 6	960716_044500_0060	1 4 5
960716_004500_0299	2 1 4	960716_050000_0250	1 3 6
960716_010000_0490	2 2 8	960716_051500_0460	2 2 6
960716_011500_0180	2 4 9	960716_053000_0160	2 2 5
960716_013000_0390	2 2 4	960716_054500_0350	1 3 4
960716_014500_0110	1 2 5	960716_060000_0540	2 4 6
960716_020000_0310	2 2 7	960716_061500_0249	1 2 3
960716_021500_0509	2 2 6	960716_063000_0450	1 2 3
960716_023000_0210	2 2 6	960716_064500_0169	1 1 3
960716_024500_0429	2 2 5	960716_070000_0361	2 4 7
960716_030000_0130	2 3 9	960716_071500_0549	2 1 4
960716_031500_0321	2 2 7	960716_073000_0260	1 1 2
960716_033000_0540	1 2 4	960716_074500_0449	2 1 4
960716_034500_0260	1 4 5	960716_080000_0140	2 0 3
960716_040000_0450	1 2 6		

* **T.P.A** : number of anti-**T**ank mines, anti-**P**ersonnel mines, and tot**A**l suspect objects (hits) that FLIR algorithm found in each test image.

Figure 5.3: Simulated FLIR algorithm test result.

There were 2 anti-tank (AT) and 4 anti-personnel (AP) mines in each FLIR image For the 33 test results in Figure 5.3 there were 55 anti-tank and 73 anti-personnel mines found, 44 false alarms in 172 hits. Therefore the test result can be calculated as

Anti-Tank found	Anti-Personnel found	Success hit	False alarm
83.3 %	55.3 %	64.6 %	25.6 %

Figure 5.4: Simulated FLIR algorithm success rate.

The test results in Figure 5 4 suggest that anti-tank mines can be easily detected, likely because their shapes are quite large and remain distinct from their neighbors On the other hand, ant - personnel mines are smaller and usually connected with other blobs via "bridges", thus their true shapes are altered and cannot be detected using the current classifier. Moreover, many sma l clutter spots can randomly combine with each other to form a structure that has feature characteristics closely resembling those of anti-personnel mines, thus yields more false alarms. Increasing the classification threshold will recognize more anti-personnel mines but will also include more clutter (false alarms). Currently there was no certain feature that can be used to distinguish anti-personnel mines and random structures One solution for this problem is a combination of image filter and segmenter which can entirely isolate a small mine shape from its surrounding. Also, a number of alternative segmentation algorithms outlined in section 2 4 might help to solve the problem as well.

Another factor that decreases the rate of success is that many random artifacts (clutter combined from smaller blobs) or some particular areas of a FLIR image might have greater contrast values than those of anti-personnel mines. Because LRT only picks the top "Iclip" contrast values, some anti-personnel mines will be left unconsidered. Possible corrections might be to increase the "Iclip" threshold, to scan contrast values independently in many non-overlap regions of FLIR, or to normalize the whole image before RMD processing.

6. Conclusions

At the completion of this contract, the Remote Minefield Detection Hierarchy (RMDH) was developed on two hardware platforms.

1) Network of transputers:

Modules NSRR, LRT and LRFE were implemented and tested on an architecture of vector processors (VP TRAMs) and transputers. The algorithms of these modules were adapted from a simulation version developed in the previous contract for FLIR imagery

Module NSRR required two vector processors, LRT needed one T805-30, and LRFE employed three T805-30 transputers. Using real FLIR images as input data, the results of these modules were verified, and they showed the processing times were slightly slow for the intended data rate of 4 frames/second.

As a new hardware platform would replace the transputer network in Phase 3, software development for the current transputer architecture halted after the completion of LRFE

2) A Pentium PC workstation:

A study by the University of British Columbia suggested that general-purpose processors, such as PowerPC or Pentium, were better suited for the RMD system, instead of using digital signal processors like SHARC or the out-of-date transputers. These modern advanced computational elements would yield much more processing power at a lower cost.

The study also indicated that some new segmentation algorithms were possibly alternatives for the current RSAM algorithm and yet would produce better results. However, the overall evaluation of these algorithms should be verified on the new hardware platform. Also, a slightly modified RSAM could be done to overcome the undesired outcomes in the present RMD system

As a consequence of that study, a network of five Pentium IV PCs, interconnected via gigabit Ethernet cards, was built in Phase 3 It was planned to run the RMD program on the main workstation, which would call and share data and work load with the other four substations in client/server fashion within the private gigabit network. It was planned to write the code in C/C++ and utilize the Microsoft Foundation Class for windows programming.

However, because of the limited time, only modules NSRR, LRS, LRFE and LRC were coded in C and intended to run in DOS mode; the other modules were still coded in the IDL language. The whole RMD program would run in Windows OS on a single Pentium IV 1.8 GHz PC The IDL package is currently used to provide GUI services and module integration

The RMD test results were impressive: a single Pentium IV 1 8 GHz PC could analyze a FLIR image in typically 1 18 second This processing time could be much improved toward satisfying

the real time speed requirement of 0.25 second / frame by carrying out suggestions given in section 7 (Recommendations for Future Work).

Lastly, a simple expert system was also designed in this contract. Because of a lack of availability of mine expert personnel during the time frame of the project, only an entry level of expert system shell was accomplished.

7. Recommendations for Future Work

Although the RMD test results were very impressive (a single Pentium IV 1 8 GHz PC could analyze a FLIR image in typically 1.18 s), the processing time could be much improved toward satisfying the real time speed requirement of 0 25 second / frame by:

- Eliminating the file swapping. Utilize static RAM instead, enhance the use of shared memory among the modules

- Optimizing the existing codes: Trim off debugging checkpoints and any unnecessary functions. Use a predetermined data format instead of reading in and checking data for system call parameters and data format. All global configurations should be read only once.

- In the RMD algorithm, using a single but fast PC (dual or quad processor with sufficient RAM): This model might yield faster speed than employing a network of client/server PCs It might also solve the inter-processor bottleneck problem, bring down the cost, and reduce the space occupied by equipment.

- Utilizing a straightforward, simple yet fast DOS program rather than a fully GUI featured windows program Similarly, avoiding a message-passing system model such as a windows application.

Furthermore, R&D work is needed to implement the RMD algorithm using a network of PCs as originally planned. This could not be completed in this contract because of the shortage of time This message-passing system will utilize the processing power of the available substation PCs

Another possible future task is to integrate all the RMD modules into a single windows application, which could employ windows programming techniques such as GUI, Microsoft Foundation Class, and multi threading.

Also, since the current RSAM segmentation algorithm did not yield a very accurate result for AP mines, further search for a better replacement is required Some alternative methods are suggested in section 2 4 such as those from the KUIM Image Processing System and possibly methods from Mathematical Morphology.

Additionally, further work is needed to expand the knowledge and rule bases for the existing entry level of expert system as well as to incorporate more essential functions.

Appendix A: Expert System and Its Applicability

A more detailed introduction to the Expert System was given in [16] The following are high lights of that report and some recent updates.

An expert system is a computer system, consisting of software and hardware, which attempts to mimic a human expert's thought process to solve complex problems or make decisions in a given field.

The Syntactic Pattern Recognition task of the hierarchical algorithm is an if-then-else based decision process, the type of reasoning that an expert system excels at. Therefore it is best to implement this module using expert system technology, and this unit will interact with the Syntactic Feature Extraction module for decision making. Furthermore, the top level of the RMD hierarchy involves the integration of disparate types of knowledge, including conflict resolution and whole minefield recognition. The characteristics of an expert system make it ideally suited for this task.

A.1 Expert System Shell

A suitable commercial expert system shell, *Elements Expert* from Neuron Data Co., has been chosen to use with the RMD system because it has the following significant features:

- A complete Rule-based development environment: One can quickly produce rules using a knowledge base, and this base can be updated at any given time. It can also be used as a rule server to provide services for existing applications.

- Event driven: It can work with external messages and react to them. These messages can come from an external event (such as a change to the GUI) or from an internal event (such as an action triggered by a rule).

- Problem solving strategy: Forward and backward chaining are available which can be switched on and off by choice or can be used in combination. LHS and RHS models are as well available.

- It can easily integrate with other language modules like C/C++ and Java. It also supports networking and standard relational databases, and has a cross-platform graphic user interface (GUI) builder

- It has a friendly user interface in Unix or windows operating systems. The company provides good technical support, and even custom development contracts.

An upgrade to *Elements Expert* package and staff training were purchased in phase 2. The software is intended to run on Unix platforms and can be linked to C/C++ modules.

A.2 Scope of the Knowledge Base

The knowledge base of an expert system is the collection of information that can be extracted by the expert system in pursuit of a solution to a problem. The significant feature of a knowledge base is that the knowledge base contains not only static data as in database but also relational information.

The static database consists of information derived from:

1. Physical geography of the suspect area:

- Types of terrain: rocks, marshes, swamps, bogs, flat uplands, river, desert,
- Types of vegetation: tree, shrub, herb, moss, . , and some categories such as height.
- Texture of terrain coarse, fine, medium, ultra-fine.

2 Tactical Command Control and Communication System (TCCCS) a distributed processing computer network system that consists of a large number of cells deployed throughout the corps area:

- Types of mine delivery means available from enemy forces: hand, mechanical dispenser, helicopter, artillery, .
- Enemy's tactics such as supporting both offensive maneuver (flank minefields) and the defence.
- Types and availability of mines from enemy forces: anti-tank (wooden, plastic,) and anti-personnel mines (PMN6, PFM-1, ...)
- Position of enemy's tactical group and weapons (armor, artillery), and the method of deployment to cover the minefield.
- The threat from enemy air attack
- The ability of the enemy's detection and command control and communication systems to execute an effective minefield deployment.
- Location and status of all friendly forces in the vicinity.

In addition, the database includes site-specific information supplied from the battle field personnel or head quarters command level which can be added to or written over the knowledge base before or during processing.

Relational information also includes production rules If-Then and If-Then-Else, which properly groups the facts into logic orders.

A.3 Knowledge Engineering

It was originally planned to extract minefield knowledge from two sources. First, information would be extracted from the tactical manuals used for mine laying and defence by the CF. Second, we would bring in a number of experts from the Canadian Forces to provide knowledge for the knowledge base. Two groups of personnel would be used. The first were CF Engineers, who have experience in planning and executing a minefield, as well as detecting and removing them. The second were Armoured Vehicle Commanders who have experience in identifying and avoiding likely mined areas. A set of questions would be asked of members of each group. Sample sets follow:

Sample questions for CF Engineers:

1. What type of ground do you lay a minefield in? ease of digging, concealment, vegetated versus bare
2. What are the terrain considerations? elevation, flatness, smoothness, vegetation
3. What are the tactical considerations? channeling, blocking, where enemy is situated
4. What is left on the minefield? manual laying, machine laying, airborne laying
5. What are the typical minefield patterns for patterned minefields?
6. What are typical densities?
7. What are typical mine type mixes?
8. Where do you NOT put a minefield?
9. Where would you put (by land vehicle or airborne platform) a scatterable minefield in place of a patterned?

Sample questions for Armoured Vehicle Commanders:

1. What areas are of interest for mobility?
2. What areas would you likely expect to be mined?
3. What areas would you likely expect to NOT be mined?
4. What paraphenalia do you expect to find on the minefield?
5. How do you remotely detect minefields now? What do you look for?
6. How much time do you have to make a waypoint decision?
7. If a remote sensor has identified an area as likely to have a minefield (based on what it senses), what information available to you would support or reject that assessment?

In the end, most of the information used in the expert system entry level knowledge base was obtained form the first source, since the schedule of CF experts turned out to not be compatible with the project time table.

A.4 Testing Methodology

Testing how well the expert system and the knowledge base perform is very crucial at this stage. The evaluation can be done by executing several test scenarios which the knowledge engineer (or expert) has already solved to see if the expert system generates the same answers through the same reasoning process.

The evaluation can be subdivided into two areas: static and dynamic:

1. The *static evaluation* is the testing of the knowledge base for its consistency and completeness. To test the completeness we should constantly ask ourselves, throughout the evaluation process, if more knowledge is need to be added to produce the correct answers.

2. The *dynamic evaluation* is the testing of the reasoning process and the advice given. The accuracy and the reliability of the decision are crucial. The important part of the dynamic evaluation is asking for an audit trail of a decision and comparing it against an expert's reasoning.

Appendix B: Computation of processing time for NSRR, LRT and LRFE

In the estimation of the run time of three modules, NSRR, LRT and LRFE, six functions govern the processing time; addition, subtraction, division, multiplication, comparison and iteration control (DO loop). Before presenting the computation of processing time of the modules, let us define the following quantities:

- n = edge size of an input image. $n = 2^{(5+m)}$ where $m = 0, 1, 2, \ldots$ M is the number of times required to compress an n by n image to a 32 by 32 region.
- Hit: Number of mines assumed per 32 by 1024 pixel area.
- T, T1 = constants determined by the type of transputer used (e.g. T425, T800 or T9000). For example, T1=12 for T800 [ref. 20].

We consider two types of transputers: T800 and T9000. The T800 is a 32-bit CMOS microprocessor and has a data transfer rate of 2.4 Mbytes/s. Its internal cycle time is 33 ns and the numbers of processor cycles performed by the computing functions: (+/-), (x), (÷), comparison, Do Loop are 7, 11, 17, 7 and 15.1 respectively [18]. The T9000 is also a 32-bit CMOS microprocessor but has much better performance. It is capable of transferring data up to 80 Mbytes/s. Its internal cycle time is 20 ns and the numbers of processor cycles performed by the computing functions: (+/-), (x), (÷), comparison, and Do Loop are 2, 2, 8, 2, and 3 respectively [19].

B.1 Computation of processing time of the NSRR module

The NSRR program consists of two subroutines: Rebin and Compress, and a main routine. The Compress subroutine was designed to compress the samples of image via consecutive block summations, and the Rebin subroutine was made to resize images. The main routine comprises three parts: (i) Compressing the samples of image, (ii) Computing the mean of all 4x4 and 32x32 windows, and (iii) Locating the points of maximum contrast.

The number of computations for the two subroutines Rebin and Compress can be found in Figure B.1, for the main NSRR routine in Figure B.2, and for the entire module NSRR is in Figure B.3

	REBIN	COMPRESS
+	$128\ (n/32)^2$	$3/2\ n^2$
x	$128\ (n/32)^2$	$3/2\ n^2$
loop	$73T\ (n/32)^2 + nT/32$	$T\ (n/2)\ (1+3n/2)$

Figure B.1: Computations for REBIN and COMPRESS subroutines.

Main routine	Compress an n by n image to a 32 by 32 region	+	$3/8\ n^2\ (\sum_{m=0}^{M} 1 \div 2^m)$
		x	$3/8\ n^2\ (\sum_{m=0}^{M} 1 \div 2^m)$
		Do loop	$T(n/4)(1+3n/4)(\sum_{m=0}^{M} 1 \div 2^m)$
	Compute the mean of the 4 by 4 and 32 by 32 regions	+/-	$128\ (n/32)^2 + (n/2)^2$
		/	$128\ (n/32)^2$
		x	$(n/2)^2 + (n/32)^2$
		Do loop	$73T(n/32)^2 + nT/32$
	Locate the point of maximum contrast	+	$2(n/2)^2 + 3(n/32)$
		Do loop	$(nT / 2^M)\ [1 + 9Hit + 8Hit\ (n / 2^{M-3})]$
		Compare	$n/2\ (n/32)^3$

Figure B.2: Number of computations required by the NSRR main routine

+/-	$n^2 [3/8 (\sum_{m=0}^{M} 1 \div 2^m) + 992/1024]$
x	$n^2 [3/8 (\sum_{m=0}^{M} 1 \div 2^m) + 257/1024]$
/	$128 (n/32)^2$
Do loop	$T [(\sum_{m=0}^{M} 1 \div 2^m) (n/4)(1+3n/4) + 73(n/32)^2 + n/32 + (n \div 2^M)(1 + 9Hit + 8Hit (n \div 2^{M-3}))]$
Compare	$n/2 (n/32)^3$

Figure B.3: Total number of computations required by NSRR module.

In order to have a 2K by 4K byte minefield image processed within one second in the NSRR stage, it was found that 40 T800 or 9 T9000 transputers are needed. And the maximum size of input images to each transputer is 512 by 512 when T800s are used, or 1024 by 1024 when T9000s are used. The data input time of a 512 by 512 image to a T800 is about .1 second and of a 1024 by 1024 image to a T9000 is approximately .01 second. The processing time (including the data input time) is presented in Figure B.4.

B.2 Computation of processing time of the LRT module

The program for LRT is quite small and simple. Most of the computations take place in the process of calculating the distances between points in the data array to eliminate invalid candidates. The process can be easily identified in the LRT program.

As we see from Figure B.5, the processing time of LRT is a function of Iclip (Iclip is the number of candidates from the top of the output list from NSRR) and two constants: T and T1 that are determined by the type of transputers used

Transputer type	Image's size (byte)	Number of Hits per 32x1024 region	Processing time (sec)	Number of transputers
T800	512 x 512	32	~.87	1
	2K x 4K	32	~1	40
T900	1024 x 1024	32	~.78	1
	2K x 4K	32	~1	9

Figure B.4: Processing time and number of transputers needed for NSRR

+/-	Iclip (3 Iclip + 4)/2
x	Iclip (Iclip + 1)/2
Compare	2 Iclip (Iclip + 1)/2
Do loop	T (Iclip)
While loop	T1 (Iclip (Iclip+1)/2 + Iclip)

Figure B.5: Number of computations required by LRT.

If Iclip is set to be 32 (as in our application) and if an 8 Mbytes image is to be analyzed, then the number of cycles performed by one T800 transputer is 1189018. Hence, the processing time is

$$1189018 \times 33 \times 10^{-9} \approx 04 \text{ second}$$

Note: For a T800, T1 = 12 [ref. 20]

With a data transfer rate of 2.4 Mbytes/s, a T800 can import 128 Kbytes data (see Figure 2.3) in about .013 second. In short, the total run time is approximately 0 06 second, well within the required time (1 second) Thus, 1 T800 is sufficient to handle the load in the LRT stage

B.3 Computation of processing time of the LRFE module

The length of code of the LRFE program is long because it is composed of a number of processes (≥ 10) used to compute feature components of a vector. However, each individual process is short, straightforward and independent of the others. Hence, the number of computations performed by each process can be easily calculated. A summary of the number of computations performed in the LRFE program is presented in Figure B.6

Feature	+/-	X	÷	Compare	Do loop
Mean	$\sum_{1}^{A}$				T (32 x 33)
Area	$\sum_{1}^{A}$				
Standard deviation	$\sum_{1}^{A}$	$\sum_{1}^{A}$	2		
Perimeter		P		$13 \sum_{1}^{A}$	
Maximum				$\sum_{1}^{A}$	
Minimum				$\sum_{1}^{A}$	
Centroid	$2 \sum_{1}^{A}$				
Length	13/8 P(P+1)	P(P+1)		¾ P(P+1)	½ T(3P+1)
Width	8P	6P	P	4P	TP
2^{nd} moment	$\sum_{1}^{A}$	$\sum_{1}^{A}$		$\sum_{1}^{A}$	T (32 x 33)

A and P are the area and perimeter of a region respectively.

Figure B.6: Number of computations required by the LRFE module.

If we assume in the worst case that all Iclip subimages are passed to LRFE from LRS and that each subimage has an equal number *N* of square regions to be analyzed, then the total number of the regions to be processed is *N* x Iclip.

In our particular application, if the LRFE module only processes regions whose sizes are at least 6 x 6 or 36 pixels (large enough to cover small mines), then for the worst case, a maximum of 28 regions in each 32 x 32 subimage are to be analyzed, or 28 x Iclip regions in total to be studied by LRFE. If Iclip is preset at 32, then the time needed to process all the regions by a 50 Mhz T9000 is ≈ 0.126 second, and by a 30 Mhz T800 is ≈ 0.8 second, within the required time (1 second).

In summary, the processing time at LRFE is a function of Iclip and the minimum size of a region chosen for the analysis. If Iclip and the minimum region size are 32 and 36 pixelsrespectively, 1 T800 is sufficient to handle the load. If more than one T800 is needed, a process farm configuration is suggested.

Appendix C: Running RMD Software

The following is a sample FLIR Hierarchical Algorithm simulation session running on a Unix Sparc station. Brief descriptions and displays will be given for each stage of the low and middle levels of the hierarchical algorithm. The windows version running on PC is quite similar. User's guide and notes can be found in "/net/frodo/export/home/projects/rmd/hieralgorithm/rmd-flir/readme.txt".

After changing directory to "/net/frodo/export/home/projects/rmd/hieralgorithm/rmd-flir/demo", start IDL and run the script file "rmd.pro". The console window displays:

```
IDL> @rmd
% Compiled module DAT2TDG
% Compiled module DISPLAY_RESULT
% Compiled module. WRITEMAPFILE.
% Compiled module. WRITETDGFILE
% Compiled module. DECODEIMAGE
% Compiled module: DECODEARRAY
% Compiled module· READOBJECT.
% Compiled module CREATEOBJECT
% Compiled module INITIALIZE
% Compiled module LOWRMD
% Compiled module LRASSEMBLE
% Compiled module LRT
% Compiled module MINE_CLASSIFY
% Compiled module NSRR.
% Compiled module RMD_EVENT
% Compiled module RMD_PROCESS
```

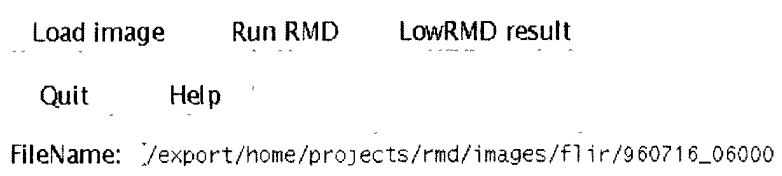


Figure C.1: Control widget.

Click "Load image" button of the above Control widget to load an 800x400 FLIR image; the selected FLIR image will be displayed:

Filter

/export/home/projects/rmd/images/flir/* dat

Directories	Files
/export/hor	960716_000000_0170.dat
/export/hor	960716_001500_0359.dat
	960716_003000_0110.dat
	960716_004500_0299.dat
	960716_010000_0490.dat
	960716_011500_0180.dat
	960716_013000_0390.dat
	960716_014500_0110.dat

Selection

/export/home/projects/rmd/images/flir/

OK Filter Cancel

Figure C.2: Select FLIR image file dialog.

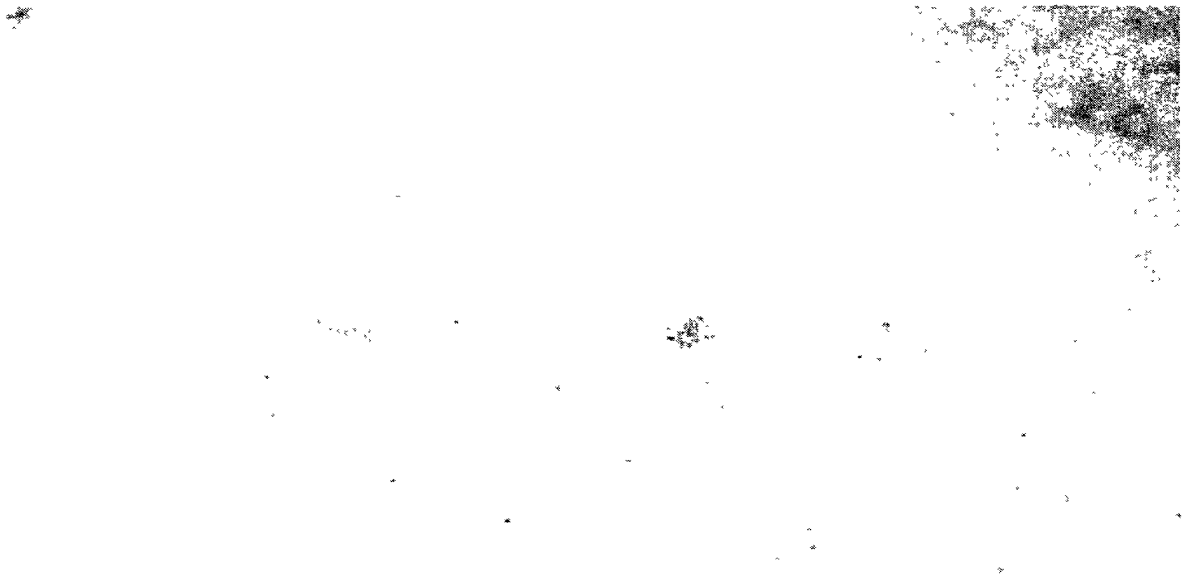


Figure C.3: A FLIR picture with 2 vivid anti-tank and 4 anti-personnel mines.

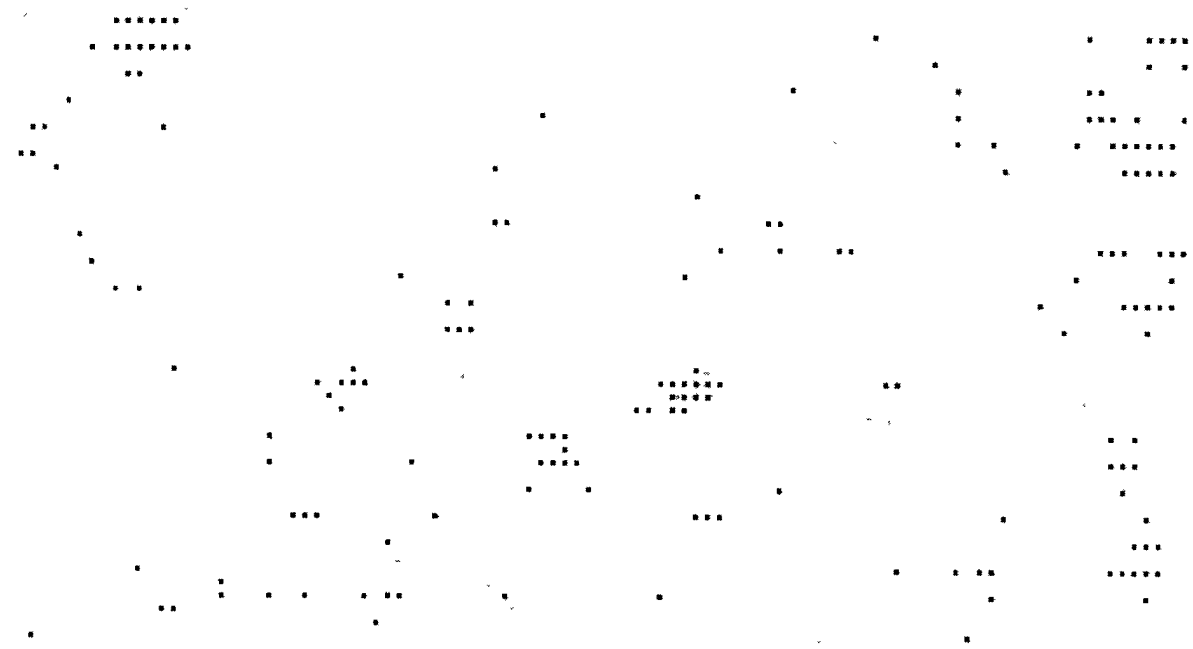


Figure C.4: A result from NSRR.

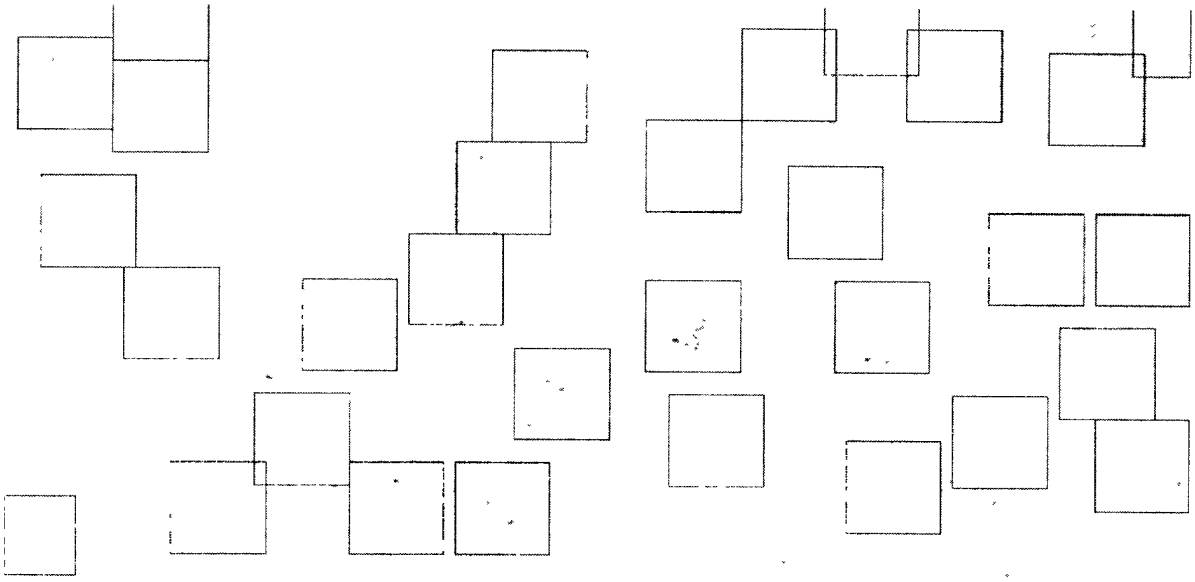


Figure C.5: An output from LRT and LRASSEMBLE.

Select the Control's "Run RMD" button to start algorithm processing. For fast operating procedure, intermediate displays resulted from NSRR, LRT and LRASSEMBLE were disabled, samples of these displays are given in Figure C.4 and C.5. The console window shows processing times in each of the modules; this timing was done on a Sparc Station 20. Then after

finishing LRC, it reports the number of Hits (objects that the algorithm presumes are mines) and Suspects (objects that were suspected by the LRT but have classification distances greater than the threshold from any known class):

```
**********************************
* Low Level RMD processing *
**********************************
* Processing NSRR
        done in      0.25207007
* Processing LRT
        done in      0.31768095
* Processing LRASSEMBLE
        done in      0.26922405
* Processing LRS
        done in      4.6891500
* Processing LRFE
        done in      9.0260390
* Processing LRC
        done in      6.0657340

**************
* RMD result *
**************
Hits, Suspects =      4        31
```

A RMD Result window appears to show the Hits in red squares and the Suspects in black squares. Left mouse click on any of these squares will display the segmented picture of the object inside along with its feature vector; while the console window reports more information of that object such as Classification data (**Class**: classification of an object according to LRC, **LRC Dist** measurement of how much an object's feature vector is different from the closest value in LRC database), **Position** (coordinates x and y of an object in an image frame), **Band** (the RMD result is an array of classified data of all suspect objects that were segmented in LRS, and "band" is the index pointing to an object's data within this result array), and **Contrast** (the difference between the intensity value of an object and the regional intensity mean value).

```
Select a region for information (right mouse to stop)

Class         = Anti-Tank
X,Y position  =    760    256
Band Contrast =     27     16.1406
LRC Dist =     0.0169823

Class         = Anti-Personnel
X,Y position  =    304    192
Band, Contrast =    17     23.4844
LRC Dist =     0.0140105

Class         = Unknown
X,Y position  =    592    224
```

Band, Contrast = 9 33.0938
LRC Dist = 0 0347126

* End of RMD result

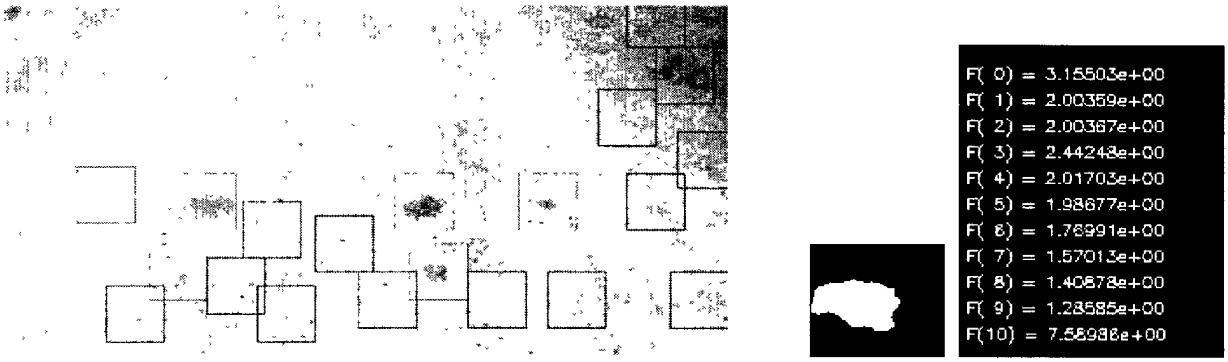


Figure C.6: A FLIR RMD result, an object's segmented image and its feature vector.

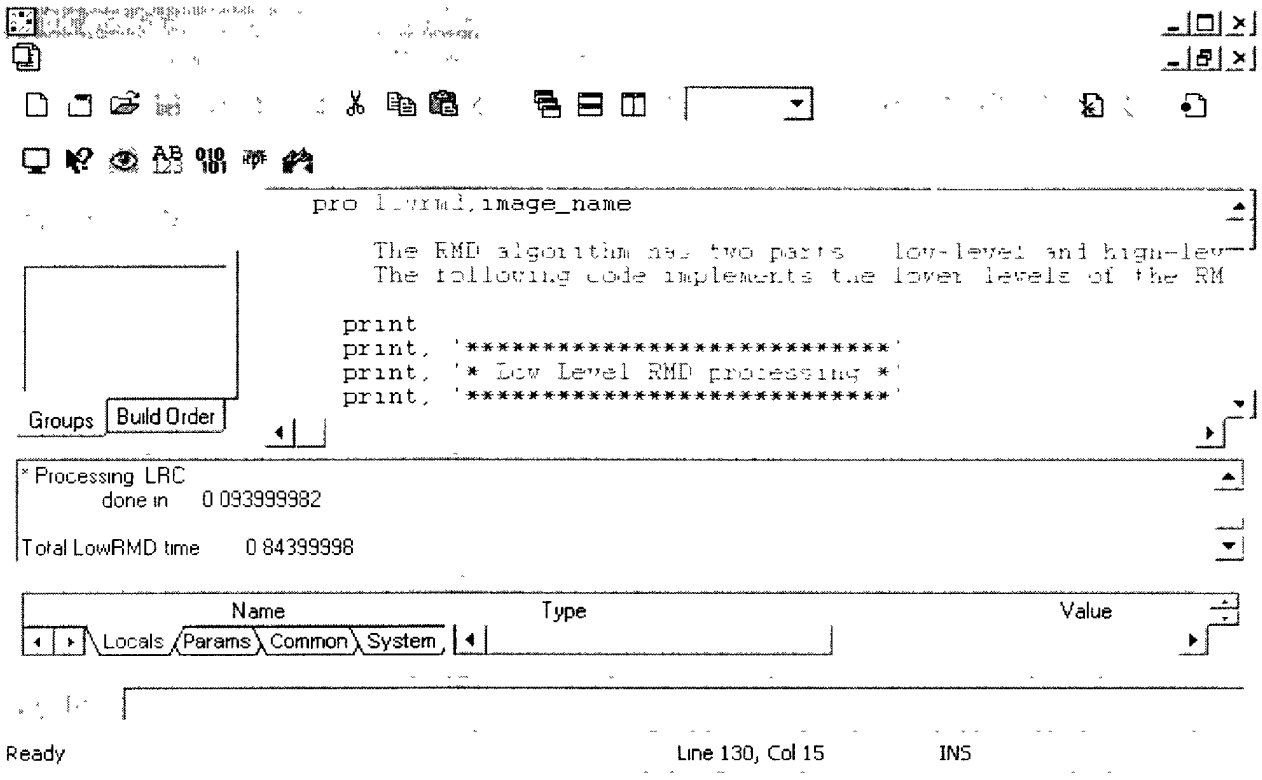


Figure C.7: PC IDL control window.

References

[1] J.E. McFee, K L. Russell, M.R. Ito, G C Stuart and Y. Das, "A Hierarchical Scheme for Analysis of Minefield Images", In *Proc. CRAD Signal Processing Symposium*, Defence Research Establishment Valcartier, PQ, Canada, June 1988.

[2] K.L. Russell and M R. Ito, "Image Analysis Application Development - Final Report", DRES Contract Report 3/90, DSS Contract File No. 01SG.97702-6-2795-A, University of British Columbia, Oct. 1989.

[3] Richard C Q. Vu, Kevin L Russell, and M.R. Ito, "Transputer Implementation of Minefield Image Analysis - Final Report", DRES Contract Report 93-11, Contract DSS file W7702-9-R121/01-XSG, University of British Columbia, March 1993.

[4] Steven B. Achal, John E. McFee, Clifford D Anger, "Identification of Surface-Laid Mines by Classification of Compact Airborne Spectrographic Imager (CASI) Reflectance Spectra", *Proc. SPIE Conference on Detection Technologies for Mines and Mine-like Targets, Vol. 2496*, Orlando, FL, USA, 17-21 April, 1995

[5] Jean-Robert Simard, Jean Dumas, L'eandre S'evigny, "Improved Landmine Detection Capability (ILDC): Evaluation of the Passive LWIR Imaging Technology to Detect Low-Metal Buried Land Mines", *Proc. of NATO IRIS Conference*, London, UK, June 1996.

[6] C G Rey and M.R. Ito, "Image Analysis Software Development - Final Report", DRES Contract Report 40/87, DSS Contract File No. 01SG 97702-R-4-9709, University of British Columbia, April 1987.

[7] R.S. Cok, *Parallel Programs for the Transputer*, Prentice Hall, New Jersey, 1991

[8] Andrew S.Tanenbaum, *Computer Networks*, Prentice Hall, Inc New Jersey, 1981.

[9] G H Ball and D J. Hall, "Isodata, an Iterative Method of Multivariate Analysis and Pattern Recognition", *Proc. of the IFIPS Congress* , 1965.

[10] E Ruspini, "A New Approach to Clustering " *Inform. Contr., Vol 15*, pages 22-32, 1969.

[11] I. Gath and A.B Geva, "Unsupervised Optimal Fuzzy Clustering." *IEEE Trans Pattern Anal. Machine Intell* PAMI-11, pages 773-781, 1989.

[12] J C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York 1981.

[13] K. Rose, E. Gurewitz and G.C Fox, "A Deterministic Annealing Approach to Clustering." *Pattern Recog. Lett , Vol. 11*, pages 589-594, 1990.

[14] F. R. Fromm and R. A. Northouse, "Some Results On Non-Parametric Clustering of Large Data Problems." In *Proc. 1st Intnl. Jt. Conf. Pattern Recognition*, pages 18-21, 1973.

[15] Yiu-fai Wong, "Clustering Data by Melting." In *Neural Computation, Vol. 5*, pages 89-104, 1993.

[16] R.C.Q. Vu and Mabo R. Ito, "System Integration Study of a Hierarchical Minefield Image Analysis Algorithm - Final Report", DRES Contract Report 95-26, DSS file W7702-1-R286/01-XSG, University of British Columbia, September 1995.

[17] J. McFee, K. Russell, M. Ito and R. Vu, "An Algorithm for Real-Time Detection of Minefields in Monochromatic Airborne Imagery", *Proc. SPIE Conference on Detection Technologies for Mines and Mine-like Targets, Vol. 2496*, Orlando, FL, USA, 17-21 April, 1995, pp. 594-605.

[18] J. McFee, S. Achal and C. Anger, "Scatterable Mine Detection Using a CASI", *Proc. of the First International Airborne Remote Sensing Conference - Volume 1*, Strasbourg, France, 11-15 September, 1994, pp. 587-598.

[19] S. Duong and Mabo R. Ito, "Implementing and Testing a Real-Time Hierarchical Mine Detection Algorithm- Final Report", DRES Contract Report CR 2000-52, DSS file W7702-4-R486/01-XSG, University of British Columbia, March 1999.

[20] SGC-THOMSON Microelectronics Group. *The Transputer Databook*, 2nd edition, 1989.

[21] SGC-THOMSON Microelectronics Group. *The T9000 Transputer Products Overview Manual*, 1st. Edition, 1991.

[22] James Ng, "Alternative hardware implementation of the Remote Minefield Detection System", University of British Columbia, August 2000.

UNCLASSIFIED

(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATOR (the name and address of the organization preparing the document Organizations for who the document was prepared e g Establishment sponsoring a contractor s report, or tasking agency, are entered in Section 8)
University of British Columbia, Vancouver, B C V6T 1W5

2 SECURITY CLASSIFICATION
(overall security classification of the document, including spe ial warning terms if applicable) UNCLASSIFIED

3 TITLE (the complete document title as indicated on the title page Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title) Extensions to Real-time Hierarchical Mine Detection Algorithm – Final Report (U)

4 AUTHORS (Last name, first name, middle initial If military, show rank, e g Doe, Maj John E)

Duong, Sinh and Ito, Mabo

5 DATE OF PUBLICATION (month and year of publication of document)
August 2000

6a NO OF PAGES (total containing information, include Annexes, Appendices, etc) 55

6b NO OF REFS (tota cited in document) 22

7 DESCRIPTIVE NOTES (the category of the document, e g technical report, technical note or memorandum If appropriate, enter the type of eport, e g interim, progress, summary, annual or final Give the inclusive dates when a specific reporting period is covered)

Contractor Final Report

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development Include the address)

Defence R&D Canada - Suffield PO Box 4000, Station Main, Medicine Hat AB T1A 8K6

9a PROJECT OR GRANT NO (If appropriate, the applicable research and development project or grant number under which the document was written Please specify whether project or grant)
12mg02

9b CONTRACT NO (If appropriate, the applicable number unde which the document was written) W7702-9-R750

10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity This number must be unique to this document) N/A

10b OTHER DOCUMENT NOs (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11 DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors, further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors, further distribution only as approved
( ) Distribution limited to government departments and agencies, further distribution only as approved
( ) Distribution limited to defence departments, further distribution only as approved
( ) Other (please specify) Distribution limited at source

12 DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document This will normally correspond to the Document Availability (11) However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected) UNLIMITED

UNCLASSIFIED

13 ABSTRACT (a brief and factual summary of the document It may also appear elsewhere in the body of the document itself It is highly desirable that the abstract of classified documents be unclassified Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C) or (U) It is not necessary to include here abstracts in both official languages unless the text is bilingual)

The Threat Detection Group (TDG) at Defense R&D Canada - Suffield has undertaken a research program on the feasibility of remote sensing of minefields. One of the projects is the Remote Minefield Detection (RMD) hierarchical algorithm originally developed for single band active airborne infrared imagery (AAII) and later adapted for vehicle mounted passive infrared imagery (FLIR) The objectives of this contract are: (i) to implement the RMD hierarchical algorithm on a down- scaled version of transputer architecture for real-time mine detection on FLIR imagery, (ii) to develop the high and top levels of the algorithm including expert system and knowledge base, and (iii) to research the possibility to upgrade the current hardware platform to modern advanced computational elements Phase 1 and 2 of the contract focused on the implementation of Low and Middle levels of a real-time RMD system on a transputer network to detect mines in real FLIR images. The work first described the hardware requirement for each module of the algorithm, then outlined the software development, and finally presented some preliminary test results Although transputers were attractive computing elements 15 years ago when this project started, they have become obsolete quickly today Thus a decision has been made by the Project Authority to implement the RMD hierarchical algorithm on the new hardware platform, namely a network of Intel Pentium-class processors, which was recommended by a study from the University of British Columbia. Phase 3 of the contract involved building that PC network, and developing and testing the new software version.

14 KEYWORDS DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document They should be selected so that no security classification is required Identifies such as equipment model designation trade name military project code name, geographic location may also be included If possible keywords should be selected from a published thesaurus, e g Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

hierarchical algorithm, remote minefield detection, computer architecture, distributed computation, real-time

Defence R&D Canada

Canada's leader in defence
and national security R&D

R & D pour la défense Canada

Chef de file au Canada en R & D
pour la défense et la sécurité nationale

DEFENCE R&D DÉFENSE

www.drdc-rddc.gc.ca