ARRAY PROCESSORS FOR IMAGE ANALYSIS

W.M. Strome*, D.G. Goodenough*, K.B. Fung*, and A.J. Baillie°
*Canada Centre for Remote Sensing
°Canadian Astronautics Limited
CANADA
Commission II, Working Group III

ABSTRACT

This paper describes some of the factors which must be considered in implementing typical image analysis algorithms on array processors. Some of the architectural and data flow problems will also be discussed.

INTRODUCTION

The rediscovery of a fast, accurate method of computing the discrete fourier transform, the Fast Fourier Transform (FFT) by Cooley and Tukey (1965) coupled with technological advances and cost reductions in digital computing hardware, led to the development of special purpose array processors in the late 1960's and early 1970's. The primary application for these processors was to support the need to perform large numbers of convolutions in seismic data in support of petroleum exploration. Similar analysis was required for the processing of acoustic data for military applications. The early processors were machines designed specifically to perform convolutions, to compute the FFT, or both. They were usually fixed point machines or "block floating point". In the latter case, a single exponent was used for all the data in an array.

In the mid 1970's, the availability of more sophisticated integrated circuits with powerful on-chip arithmetic capability made it practical to build array processors which were programmable and much more versatile than their predecessors, although many of these were still optimized for the FFT operation.

During the same time period, a few general purpose "super computers" appeared on the scene. Most of these high speed processors use a technique known as "pipe-lining" to increase the effective speed of computation on suitably structured problems beyond that possible with traditional computer architecture.

Since the early 1960's, an alternative architecture has been explored by several groups. This is the use of an array of identical processors operating in parallel, or parallel processors.

To meet the need to process very high volumes of image data in support of resource management applications of remotely sensed data, scientists and engineers have turned to the use of array processors to increase the speed of analysis.

ARRAY PROCESSORS

The term "array processor" is used to describe special purpose computing devices which have been optimized to perform highly repetitive operations on large volumes of data. In this section, we will discuss only those systems which perform their operations sequentially.

The secret of efficient, repetitive computation on very large arrays of data is a combination of two architectural features which most of these systems have in common. The first is "pipe-lining". This is a technique whereby each standard operation is broken down into several elemental steps. After the first computational step is performed on the first set of data, the results are moved to the second stage while the first step begins on the second data set. In this way, if the time required for the basic logic circuits used to construct the machine is T, and the pipeline has 5 stages, for a very large array of data, the effective time for operation is approximately T/5.

The second commonly found architectural feature is an independence of data and control information. In traditional computers, instructions are fetched from the main memory and interpreted. The data are then read from the memory or internal registers, an operation performed and the results stored. The typical array processor has a separate control computer (usually with its own, separate memory) which directs the high speed computational elements as to which operations it is to perform on an incoming stream of data. Usually the data flow (sequencing of addresses in data memory, etc.) is also controlled separately.

If the types of operations which can be performed by the high speed pipeline processor are sufficiently flexible, and the process required can be structured to permit sequential processing of large amounts of data in the same manner, considerable efficiency can be achieved, resulting in speed improvements of one or more orders of magnitude over that of a general purpose computer using the same circuit elements.

The speed of an array processor is not so much derived from that of the arithmetic logic units. The principal factor is that the operations are so highly repetitive that the control can be performed in parallel. Thus, in one operation, a few additions and one or two multiplications are performed in parallel, and these are repeated time after time. Thus, for example

$$Z_i \leftarrow a_i x_i + b_i y_i$$

may be performed in one cycle, whereas, in a computer, one might have to perform a sequence such as:

```
L1:    $IZ_i \leftarrow I_i + 1$
       if $r_i > n$, go to L2

       $IA \leftarrow IA+q$
       $IB \leftarrow IB+p$
       $IX \leftarrow IX+u$
       $IY \leftarrow IY+v$
       $IZ \leftarrow IZ+w$
       $B \leftarrow b(IB)$
       $A \leftarrow a(IA)$
       $B \leftarrow B.y(IY)$
       $A \leftarrow A.x(IZ)$
       $Z \leftarrow A+B$
       $Z(IZ) \leftarrow Z$
       go to L1
```

```
L2:    continue
```

where Z_i are the results, a_i, x_i and z_i are the operands for the ith operation, IA, IB, IX, IY and IZ are registers which may be used for indexing, and B, A, and Z are registers for temporary storage. This would require 14 operations, and at least 19 memory cycles. The array processor is usually set up with separate memories so that the same sequence is effectively executed in a single machine cycle and only one or two high-speed memory cycles. Of course, there is considerably more overhead in the array processor involving the initial storage of the data in its own memories and setting up of the address calculator processor. Thus, the effective speed is only approached when the same sequence of operations is performed on a large number of data points. This is usually the case in image analysis or in the computation of FFT for many thousands of points.

COMMERCIAL ARRAY PROCESSORS TO AUGMENT GENERAL PURPOSE COMPUTERS

One approach to the use of an array processor for image analysis is connecting a commercial array processor to a general purpose computer. This software performs the various enhancement and classification functions required for the analysis of remote sensing image data. The philosophy employed in the connection of the array processor is that of off-loading the central processing unit (CPU) for those compute-bound operations involving the highly repetitive computations typically performed in image analysis. The interface to the general purpose computer permits the array processor to access directly the main memory as required, to convert the fixed- and floating-point formats of the host CPU to the internal floating-point format of the array processor and to read out the control information supplied by the CPU. It is possible for the operating system and mathematical library of the host to be modified so that all operations of a highly repetitive nature (FFT, matrix operations, classification and image enhancement) will occur within the array processor in a manner which is completely transparent to the user. This results in a marked increase in real-time speed of operation and a drastic reduction in CPU time used for such analysis. As library routines are modified, more of the computer-bound functions can be transformed to the array processor, thus relieving the CPU for other types of work. The commercial array processor is optimized for FFT computations, but can provide substantial improvement in the execution times of various classification algorithms.

SPECIAL-PURPOSE ARRAY PROCESSORS

CCRS has developed a facility designed specifically for image analysis (Goodenough, 1977). The heart of this is the General Electric Image-100 which includes a powerful general-purpose minicomputer, an interactive colour display system and special-purpose hardware to compute rapidly n-dimensional histograms and to perform supervised parallelepiped classification. A special-purpose image analysis processor (IAP) has been designed to augment this system by rapidly performing the highly repetitive, and very time-consuming tasks normally performed in software (Goodenough, 1978). The arithmetic pipe-line was optimized for classification with the maximum-likelihood decision rule. However, it was also provided with the special instructions and data paths required for efficient performance of the FFT.

One of the more frustrating delays which normally occur in a typical minicomputer-based image analysis system is that of transferring the image data between the mass storage medium, either magnetic tape or disc, and the refresh memory of the colour display system. This is especially so when the transfers involve picture element (pixel) replication or decimation as that involved in scale changes. The IAP has been designed to perform these functions most efficiently and without loading the host CPU.

An important problem in spatial image analysis is that of performing a two-dimensional FFT, which is computed by first taking a one-dimensional FFT of all the rows (or columns) of an array, and then performing a second one-dimensional FFT on the resulting columns (or rows). If all the data are in the random access memory (RAM) of the processor, this presents no particular problem. However, for large arrays, it is not practical to provide a sufficiently large RAM; the array must be stored on a device such as a disc, and operated upon in segments. Usually these are one row or column at a time. It is obvious that if the data are stored on a disc, they may be accessed efficiently either by row or by column, but not both. The IAP input/output (I/O) processor is designed to perform automatically the necessary disc accesses to retrieve or store the data in the 'inefficient' mode without impact on the host CPU or IAP processing efficiency.

In designing the IAP, an important cost consideration was the total number of bits of data to be carried in the arithmetic unit and internal storage. Also, the ultimate speed of the unit, for a given type of logic component, is reduced as the number of bits increase. A study was conducted to determine the minimum word size which would be acceptable. Simulations, with various word sizes, were carried out for both maximum-likelihood classifications and two-dimensional FFT. It soon became obvious that the error propagation in the FFT calculations is much more serious than with maximum-likelihood estimates. The criterion used was that a two-dimensional forward and inverse FFT of 512x512 points should agree within 1% of that obtainable using 32-bit floating point. It was found that an 18-bit floating-point format (5-bit exponent and 13-bit mantissa) met the design goal of less than 1% error. The error in likelihood calculations was always less than 0.1%.

A block diagram of the IAP is given in figure 1 (Goodenough, 1978). It is interfaced to the host CPU and to the second port of a large two-port moving disc drive through a 1000-word first-in first-out (FIFO) buffer memory which permits each device to operate asynchronously of the others. The IAP is also directly connected to the Image-100 processor and display memory through a high-speed (10 megabytes/second) 4096-byte line buffer memory (LBM), which matches the 10MHz data rate of the recirculating display memory. The IAP thus has direct access to the display storage and the special-purpose Image-100 processor ratioing and matrix multiplication hardware.

The programmable control computer (CC) controls the initialization of registers which, in turn, control the other elements of the IAP. The CC, using 18-bit instructions stored in the program memory (PGM) performs any format conversion required of the data and is used to transfer data from one memory to another. The program memory (PGM) consists of 256 words of read-only memory to store interrupt and bootstrap programs and 1k 18-bit words of RAM, which may be loaded from the host.

The pipe-line processor (PLP) is an 18-bit floating-point-arithmetic unit composed of four major sections, the pipe-line memory system (PLMS), the scratch pad memory system (SPMS), the pipe-line (PL) and the output buffer. Once the PLP has been initialized by the CC, the PLP can perform independent computations. Thus, the CC and PLP can operate in parallel. The PLP has an effective throughput rate of 100 ns. The PLMS includes three 18-bit 1k RAM, marked as PLM1 to PLM3 in figure 1. For maximum-likelihood, four-feature, eight-class calculations, for example, PLM1 would contain 256 four-feature pixels, PLM2 would store eight four-feature means, and PLM3 would have eight inverse covariance matrices plus the decision rule threshold. Complex address calculators in the PLMS ensure that the address sequence required for FFT is performed correctly. Data from the PLMS are called for by the SPMS which then delivers the data to the pipe-line according to the sequence of instructions held in the scratch pad (SP) memories. The SP can hold up to 32 16-bit instructions plus eight 18-bit data words. Each SP program instruction is executed in 100 ns so that data may be passed to the pipe-line at that rate. The data from SP3 is delayed in order to arrive at the second multiplier coincident with the output from the first multiplier.

The output of the second multiplier is a 23-bit floating-point number with 5 bits for the exponent and 18 bits for the mantissa. To achieve the required speed in the accumulator, it was found necessary to convert from a floating-point to a fixed-point (47 bits) number (UFLT), accumulate, and then convert back to an 18-bit floating point number (FLT). When the required number of terms have been summed, the computed likelihood value, for example, is passed to the output buffer. This simplifies the programming of the PLP since the output buffer thresholds the maximum-likelihood value, and outputs

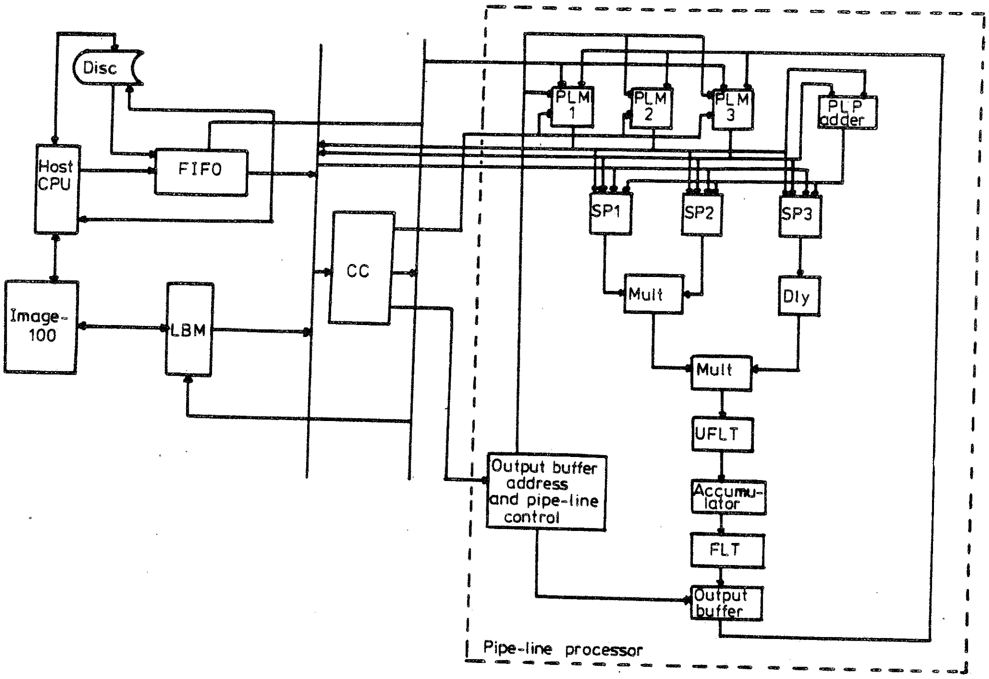


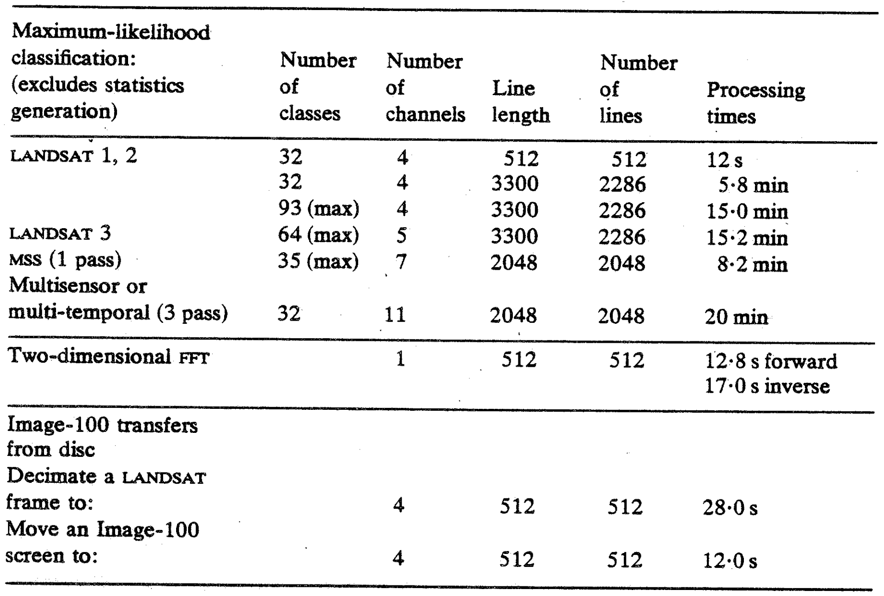
Figure 1. Block Diagram of CCRS Image Analysis Processor (IAP)

the resulting class number (16-bit integer) and likelihood value to PLM1. The class number may be coded as an integer or with one bit set corresponding to the class. In the FFT mode, the real and imaginary values are returned to PLM1 and PLM2, respectively. Thus, as data are flowing our of the PLMS, results are flowing into them. When the data have been processed, the array of results are sent to disc, the Image-100 or to the host CPU memory.

The size of memories in the PLP restrict the number of features and classes which may be accommodated in one pass for maximum likelihood. The PLP can calculate the maximum-likelihood function for M classes and N features at a rate of $10^{-7}M(N_T+1)$ seconds per pixel. Program and input/output (I/O) overheads increase this time for large images. Table 1 summarizes the processing rates achieved by the CIAS with the IAP for maximum-likelihood classification, two-dimensional fast Fourier transforms, and transfers from the large disc to the Image-100 solid-state memory. The parallel processing ensures that these rates are achieved, within 10%, even under full system load. The maximum number of classes M_{max}, for one pass is limited by N and by the size of the PLM. Some examples are listed in Table 1.

To classify a full scene, an N-feature 512x512 subscene is first analyzed interactively. The parameters, statistics and input video files are then read by the IAP and the whole scene classified.

Table 1. CIAS/IAP Processing Times



Maximum-likelihood classification: (excludes statistics generation)	Number of classes	Number of channels	Line length	Number of lines	Processing times
LANDSAT 1, 2	32	4	512	512	12 s
	32	4	3300	2286	5·8 min
	93 (max)	4	3300	2286	15·0 min
LANDSAT 3	64 (max)	5	3300	2286	15·2 min
MSS (1 pass)	35 (max)	7	2048	2048	8·2 min
Multisensor or multi-temporal (3 pass)	32	11	2048	2048	20 min
Two-dimensional FFT		1	512	512	12·8 s forward 17·0 s inverse
Image-100 transfers from disc					
Decimate a LANDSAT frame to:		4	512	512	28·0 s
Move an Image-100 screen to:		4	512	512	12·0 s

PARALLEL PROCESSORS

While the array processors of the previous section provide efficient execution of array operations, they do so in a serial fashion, in that only a single point in each array is operated upon at a given time. In parallel processing, many operands are dealt with in parallel. Perhaps the classic example of a parallel processor is that of the ILLIAC IV (Barnes et al, 1968), which is a parallel array of 64 general-purpose computers or processing elements (PE). Each PE executes the same instruction on different data items under control of an executive computer or control unit (CU) which decodes the program instructions and distributes them in parallel to the PE. The CU handles all program branching and subroutine linkages. The CU can also select specific PE to remain idle during any instruction period, or a PE may remain idle on the basis of the last calculated result.

Each PE has its own memory of 2048x64-bit words which is used for both program and data storage. The total memory may be thought of as an array of 64 columns (one for each PE) by 2048 rows. The CU reads programs row-wise and may access any location in memory, while each PE may directly access its own column. To obtain data from some other column, it must first be read into its own PE, then routed to the PE requesting it through an interconnection network. Each PE, numbered 0 to 63, is connected to four others; PE(i) is connected to PE(i±1mod64) and PE(i±8mod64). Figure 2 shows the organization of the ILLIAC IV, including the interconnection structure.

Under optimum conditions, the ILLIAC IV has an effective execution speed of about 200 million instructions per second (MIPS), and can classify a 3300x2286 pixel four-channel image into 30 classes in about 3-5 min (Hord, 1978). This is similar to the 5-8 min required by the IAP for the same size image and 32 classes. A typical commercial array processor would require approximately one-half to two hours (extrapolated from Bradford (1978): 512x512 pixels x 4 channels and 8 classes require 56 s).

Figure 3 shows a 4x4 schematic of the massively parallel processor (MPP) proposed by Fung (1977). A 128x128 version has been built. The MPP consists of a two-dimensional array of processing units, each of which contains: basic memory elements (M_i); sliding units (s) which transfer data to adjacent PU and eventually to and from the external environment - this, therefore, acts as the input/output (I/O); logical processors (P) which can transfer data to adjacent PU, and can also perform logic operations such as AND, OR and XOR; counter/shift registers (C) which perform arithmetic operations, including add, subtract and multiply; and masking devices (G) which control the mode of operation of the P and C. Each PU has a 1-bit data bus, and except for the arithmetic unit, each device stores only one bit. The least significant bit of the register is available to the data bus. The total register contents may be incremented or shifted left or right.

Each PU operates in a bit-serial fashion, so that any arbitrary word length may be used. Because of the bit-serial nature, each PU is relatively slow, but the massively parallel structure should enable it to operate at very high effective speeds. The sample classification of 3300x2286 pixel four-band images into 30 classes will require about four minutes.

LANDSAT DIGITAL IMAGE ANALYSIS SYSTEM (LDIAS)

CCRS is developing a new image analysis system (LDIAS) which can analyze a full Thematic Mapper scene into 32 classes in at most 8 hours while enabling the integration of geocoded data. The broad architecture of LDIAS is shown in Figure 4. There are three D.E.C. computers, two VAX 11/780's and a VAX 11/730. The image displays, from DIPIX and Gould Deanza, contain processing hardware for spatial filtering and edge enhancement, spectral enhancements, and parallelepiped classification. The Intergraph map displays show digital terrain models and digital maps. With these displays, one can rapidly select ground control points for image registration or rectification.

A Thematic Mapper scene, corresponding to 185km by 185km, occupies a 500 megabyte disk. Training areas are selected based on maps, photos, or manually off images. Large area classification is carried out in the Fast Multidimensional Processor System (FMPS). FMPS includes a VAX 11/730 control computer, dual-ported disks with VAX #1, a STAR-100 array processor, and a high-speed preprocessor unit (PPU). The PPU can perform table lookups and histogram acquisition at disk access speeds. It can also produce a parallepiped classification of a 21-channel image into 256 classes. Class

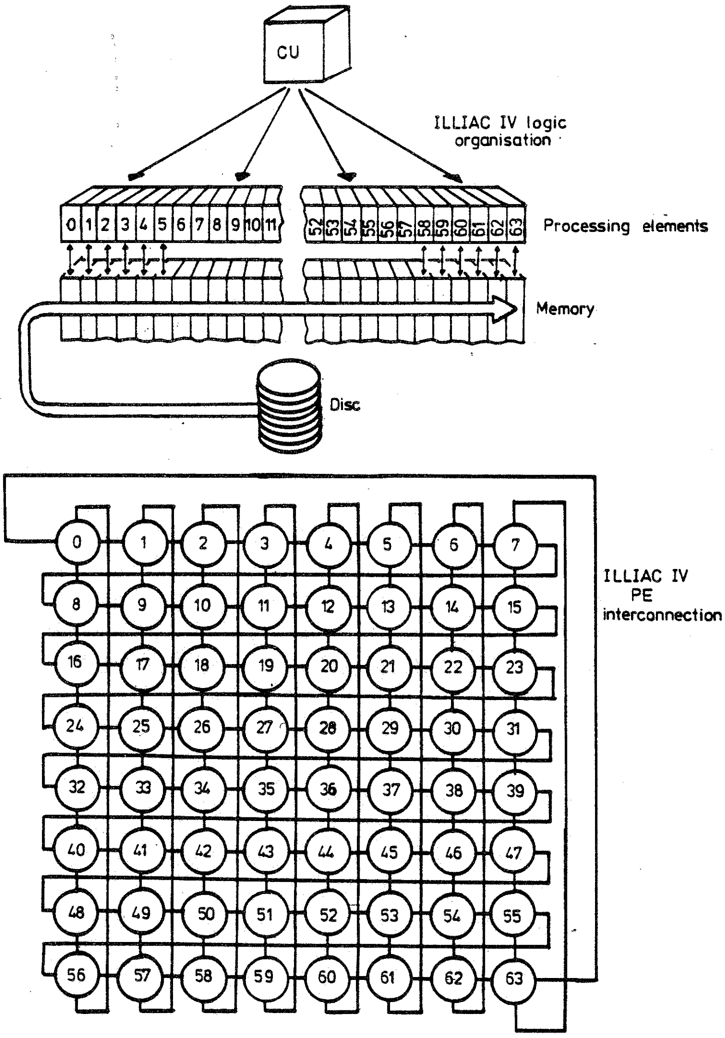


Figure 2. Organization of ILLIAC IV

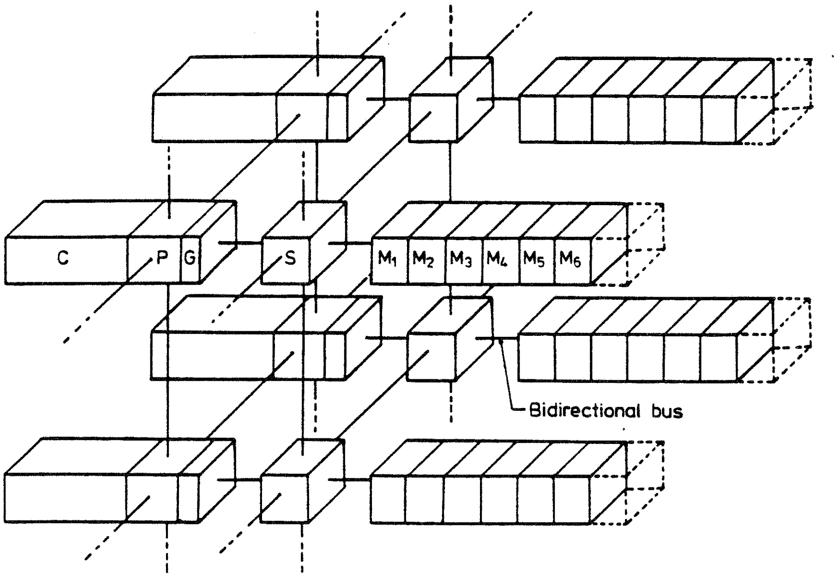


Figure 3. Massively Parallel Processor

overlaps can be resolved in the STAR-100 using maximum likelihood on pixels with class confusion or overlap. The resulting classification can be communicated to VAX #2 for conversion to polygons from a raster grid file. The polygon file can be down-loaded over an automatic communication link to compatible geographic information systems.

As can be seen from Figure 4, LDIAS is a complex system. In order to simplify its operation, CCRS is investigating the use of expert system approaches to simplifying the operational interactions for the 600K lines of code in this system. Throughput has been sacrificed to some degree to enhance flexibility of the system for changes in algorithms and applications.

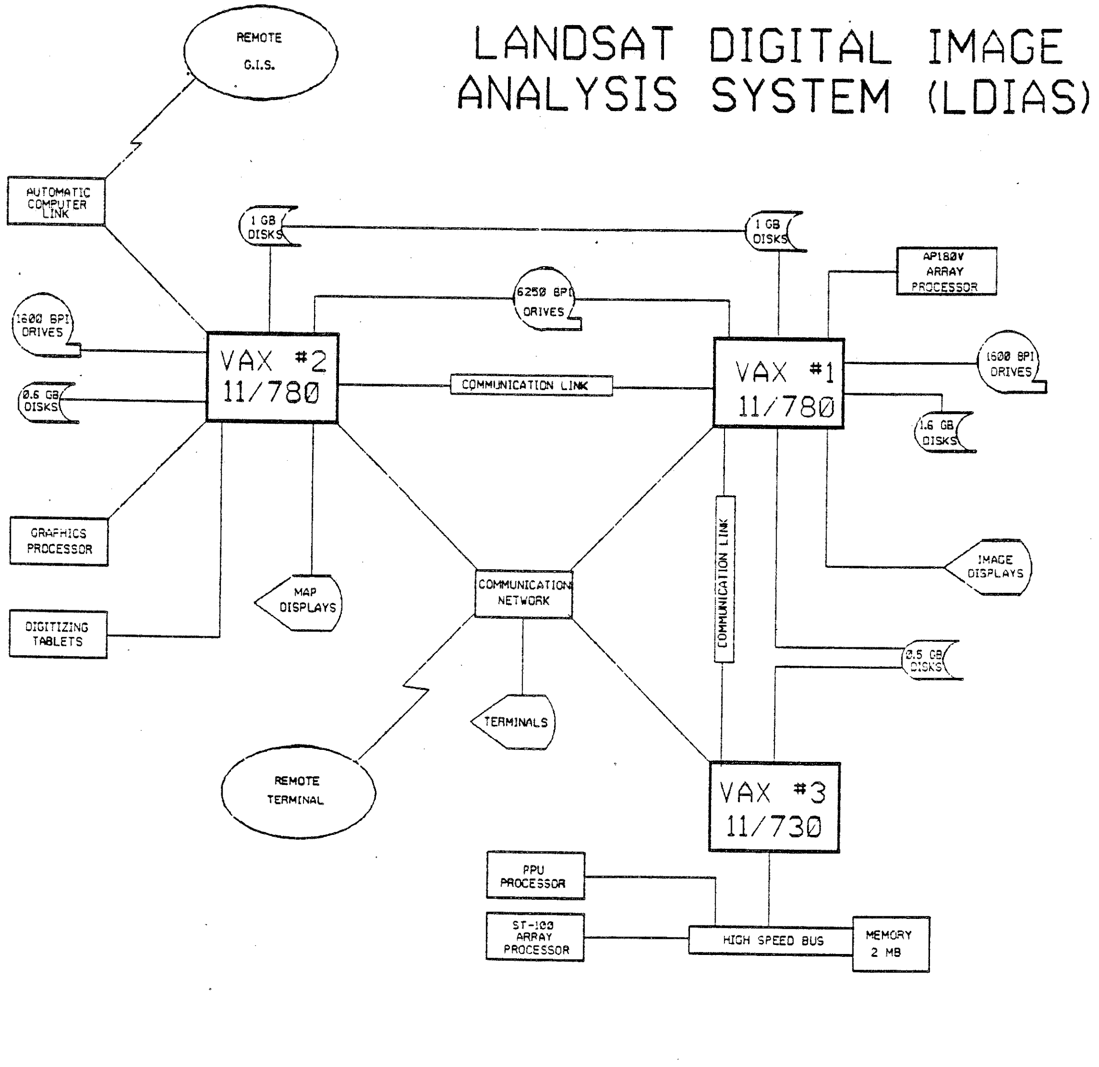


Figure 4. LANDSAT Digital Image Analysis System (LDIAS)

CONCLUSIONS

Image analysis requires the performance of highly repetitive calculations on very large amounts of data. A typical LANDSAT-5 TM image, acquired in 25 seconds, is represented by 500 million bytes and requires two 2400 foot reels of computer tape at 6250 bpi recording density. Newer satellites will provide even more data. To analyze these data effectively, special-purpose processors will be used more frequently. Those described reduce the computation times over standard large-scale, general-purpose computers or the more powerful minicomputers by factors ranging from 30 to 200.

REFERENCES

Armstrong, W.M. and Gecsei, J., 1978, Architecture of a Tree-Based Image Processor, Dépt. d'informatique, Université de Montréal, Publ. No. 291.

Barnes, G.H. et al, 1968, The ILLIAC IV Computer, IEEE Trans. Comput. C-17, 746-57.

Bradford, L.H., 1978, Application Systems for Imaging Data, Proc. Floating Point Systems Users Group, New Orleans.

Cooley, J.W. and Tukey, J.W., 1965, An Algorithm for the Machine Computation of Complex Fourier Series (FFT), Math. Comput. 19, 297-301.

Fung, L., 1977, A Massively Parallel Processing Computer, Proc. Symp. on High Speed Computers and Algorithm Organization, University of Illinois.

Goodenough, D.G., 1977, The Canada Centre for Remote Sensing's Image Analysis System (CIAS), Proc. 4th Canadian Symp. on Remote Sensing, Quebec City.

Goodenough, D.G., 1978, Programming Hardware for Remote Sensing Image Analysis, Proc. AFIPS National Computer Conf., Anaheim paper V.47.

Hord, R.M., 1978, Digital Processing of LANDSAT Imaging on ILLIAC IV, Institute for Advanced Computation, Alexandria, Va., Report.

Salomonson, V.V., 1978, LANDSAT-D, A Systems Overview, Proc. 12th Int. Symp. on Remote Sensing of Environment, Quezon, Philippines.